Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`

2. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4

3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4

4. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

5. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`

6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux

7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `1 2 3 1 2`

8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 4`
   - ☐ `j = 0`

10. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `** *** **** *****`

11. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `kwrite TP4`
- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `mkdir TP4`

12. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n",x y);`

13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`

14. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define taille = 3`

16. Le langage C est un langage

- ☐ lu, écrit, parlé
- ☐ composé
- ☐ interprété
- ☐ compilé

17. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`

18. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule manquant
- ☐ une accolade en trop
- ☐ une accolade manquante
- ☐ un point-virgule en trop

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source
- ☐ qu'on veut changer alétoirement de fond d'écran

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
2. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
3. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
4. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
5. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique
   - ☐ l'édition de liens
6. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
10. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 0
- ☐ 16
- ☐ 8

12. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ ouvir un bureau partagé (common desktop)
13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ la variable x vaut 16
    - ☐ le programme affiche x
    - ☐ le programme affiche `****`
14. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
15. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
17. L'ordonnancement par tourniquet permet :
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ d'afficher des ronds colorés à l'écran
18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`
19. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int %d;`
    - ☐ `int k;`
    - ☐ `int loop n;`
20. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ interprété
   - ☐ composé

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16

3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc

4. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
   - ☐ créer un fichier texte

5. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une variable non déclarée

6. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 5`

8. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1

9. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`

10. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte

11. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Majeur Mineur`

12. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 1 2 3 4

14. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 3
    - ☐ 16
    - ☐ 6
    - ☐ 20

15. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`

17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`

18. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ init
    - ☐ begin
    - ☐ main

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`

20. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 6
    - ☐ 42
    - ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose

2. Le langage C est un langage
   - ☐ interprété
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé

3. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `***** **** *** ***`
   - ☐ `** ** ** ** ** **`
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`

4. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
   - ☐ une accolade en trop

5. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

6. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`

8. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ begin
   - ☐ include
   - ☐ init

9. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 4
   - ☐ 0
   - ☐ 16

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ ne comportent aucune erreur

12. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source

13. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
15. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 20
    - ☐ 16
    - ☐ 3
    - ☐ 6
16. Quel est l'opérateur de différence en C :
    - ☐ `<>`
    - ☐ $\neq$
    - ☐ `!=`
    - ☐ `!`
17. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ changer de répertoire courant
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
18. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 8 6 4 2 0
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2
20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** −0,5 **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

2. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ ***** **** *** ***
- ☐ **** **** **** ****
- ☐ ** ** ** ** ** **
- ☐ ** *** **** *****

3. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `mkdir TP4`
- ☐ `kwrite TP4`

4. Le bus système sert à :

- ☐ Arriver à l'heure en cours
- ☐ Écrire des données sur le dique dur
- ☐ transporter les processus du tourniquet au processeur
- ☐ Transférer des données et intructions entre processeur et mémoire

5. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte
- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0

7. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 15
- ☐ 10
- ☐ 6

8. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ l'instruction qui met fin à un programme
- ☐ un chiffre binaire (0 ou 1)
- ☐ la longueur d'un mot mémoire

9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`

11. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé
    - ☐ interprété

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ le programme affiche `****`
    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche x
    - ☐ la variable x vaut 16

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`

15. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition

17. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant

18. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 4
    - ☐ 8
    - ☐ 0

19. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique

20. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ voir des clips musicaux
   - ☐ compiler un programme

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`

3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x y);`

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`

6. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8

7. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ init
   - ☐ include
   - ☐ main

8. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

9. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ certaines données de la mémoire de travail
   - ☐ des processus
   - ☐ en temps d'accès
   - ☐ les fichiers du disque

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"

11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
```

```
14        x = y;
15
16        ...
17    }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3

13. Le langage C est un langage
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ composé
14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`
15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 0
- ☐ 6
- ☐ 1

16. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
17. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 0
- ☐ 16

18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ le programme affiche `****`

20. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)
2. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
   - ☐ une faute de frappe dans un appel de fonction
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
4. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
5. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte
7. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
10. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 42
    - ☐ 6
    - ☐ 0
11. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
12. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade manquante
    - ☐ un point-virgule en trop
13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`

14. Le langage C est un langage

- ☐ interprété
- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ composé

15. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse harmonique
- ☐ analyse syntaxique
- ☐ analyse sémantique

16. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`

17. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `mkdir TP4`
- ☐ `new TP4`
- ☐ `kwrite TP4`
- ☐ `yppasswd`

18. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 4
- ☐ 16

20. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

2. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

3. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source

4. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ `<>`
   - ☐ `!`
   - ☐ $\neq$

5. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1

6. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien

7. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une variable non déclarée
   - ☐ un caractère interdit en C

8. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`
   - ☐ `** ** ** ** ** **`

9. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 3
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable y vaut 5

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"

12. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int %d;`
- ☐ `int loop n;`

13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `mkdir TP4`
- ☐ `kwrite TP4`

15. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'afficher des ronds colorés à l'écran
- ☐ d'entretenir l'illusion que les processus tournent en parallèle

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre

18. Le langage C est un langage

- ☐ composé
- ☐ lu, écrit, parlé
- ☐ compilé
- ☐ interprété

19. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique
- ☐ l'analyse harmonique

20. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
2. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = 4`
4. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
5. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `1 2 3 1 2`
6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
7. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 15
   - ☐ 10
   - ☐ 6
   - ☐ 0
8. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'afficher des ronds colorés à l'écran
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
11. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ voir des clips musicaux

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 0
- ☐ 8

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien

16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1

17. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 6
- ☐ 1
- ☐ 42

18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N 3`
    - ☐ `#define N =  3`

19. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ en temps d'accès
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail

2. L'ordonnancement par tourniquet permet :
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de ne pas perdre de temps avec la commutation de contexte

3. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`

4. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'édition de liens
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`

6. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
   - ☐ un point-virgule manquant

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 0 1 1 2 2`

9. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `<>`
   - ☐ `!`
   - ☐ `!=`

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`

11. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 16
    - ☐ 20
    - ☐ 6
    - ☐ 3

12. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

13. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire
14. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé
15. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
16. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6 7
19. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
20. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`

2. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire

3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 1 2 3 4

4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`

5. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 8
- ☐ 0
- ☐ 16
- ☐ 4

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0

7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche x

9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

10. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 6
- ☐ 0
- ☐ 15

11. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 3
    - ☐ la variable x vaut 5 et la variable y vaut 3

13. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ main
    - ☐ include
    - ☐ init

14. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6

15. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`

17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4

18. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible

19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 16
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le terminal affiche 8
    - ☐ le bus explose

20. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ 1 2 3 1 2
    - ☐ 0 0 1 1 2 2
    - ☐ 0 1 0 1 0 1
    - ☐ 0 1 2 0 1 2

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès
   - ☐ les fichiers du disque

2. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int %d;`
   - ☐ `int k;`

3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`

4. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 5`

6. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`

7. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `!=`
   - ☐ `<>`

8. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4

9. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1 0 1`
    - ☐ `1 2 1 2 3`
    - ☐ `0 0 0 1 1 1`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`

12. L'ordonnancement par tourniquet permet :

- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible

13. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 4`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0

15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

16. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut lancer un déboggueur

17. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`

18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8

20. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 4
- ☐ 0

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 0

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`

5. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur

6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

7. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade manquante
- ☐ un point-virgule manquant
- ☐ une accolade en trop

8. Un registre du processeur est :

- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ un composant qui contient la liste des fichiers du système

9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 8 6 4 2

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`

11. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 6
- ☐ 10
- ☐ 0

12. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ include
- ☐ init
- ☐ begin
- ☐ main

13. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`

14. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition

16. Un fichier source est :

- ☐ un document qui doit être protégé
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document illisible pour les humains
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

17. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

18. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur

19. Le langage C est un langage

- ☐ composé
- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ interprété

20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

2. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define N =  3`
- ☐ `#define N 3`
- ☐ `#define taille = N`

3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique
- ☐ l'analyse sémantique
- ☐ l'édition de liens

4. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

5. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 4
- ☐ 16

6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

7. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ retourner un bloc
- ☐ sélectionner entre deux blocs à l'aide d'une condition

8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

9. Sous unix (ou linux), la commande `cd` permet de :

- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)
- ☐ changer de répertoire courant
- ☐ détruire un fichier
- ☐ récupérer un programme arrêté avec la commande `ab`

10. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut indenter le fichier source
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut lancer un déboggueur

11. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

12. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

13. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2

14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `** *** **** *****`
- ☐ `**** **** **** ****`

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 10
- ☐ 15
- ☐ 6

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`

18. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse sémantique
- ☐ analyse harmonique
- ☐ analyse syntaxique
- ☐ analyse lexicale

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

20. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `char tableau[5];`
- ☐ `int[] new tableau(5);`
- ☐ `int tab[] = 5;`
- ☐ `int toto[5];`
- ☐ `int toto[taille=5];`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

3. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ compilé
   - ☐ composé

4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée

5. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti

6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`

7. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
   - ☐ 15
   - ☐ 0
   - ☐ 6
   - ☐ 10

8. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant

9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `1 2 3 1 2`

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6

12. Un fichier source est :

- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document de référence du système

13. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[taille=5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[5];`
- ☐ `char tableau[5];`
- ☐ `int[] new tableau(5);`

14. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut indenter le fichier source

15. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 1 2 3 4

16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

17. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse syntaxique
- ☐ analyse lexicale

18. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int %d;`

19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
            ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

20. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 0`

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4

4. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`

5. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse sémantique
   - ☐ analyse harmonique
   - ☐ analyse syntaxique

6. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Mineur`

7. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble

8. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 3
   - ☐ 16
   - ☐ 6
   - ☐ 20

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3

10. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
11. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur
12. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
13. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
14. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2 0
    - ☐ 8 6 4 2
15. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ un caractère interdit en C
16. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `!`
    - ☐ $\neq$
    - ☐ `<>`
17. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
19. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
20. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ voir des clips musicaux

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`
- ☐ `**** **** **** ****`

2. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`

3. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?

- ☐ un caractère interdit en C
- ☐ une directive préprocesseur `#include` manquante
- ☐ une faute de frappe dans un appel de fonction
- ☐ une variable non déclarée

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

5. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`

6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6

7. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte

8. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`

10. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
11. Quel est l'opérateur de différence en C :
    - ☐ `<>`
    - ☐ `!=`
    - ☐ `!`
    - ☐ $\neq$
12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
14. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2
    - ☐ 8 6 4 2 0
15. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 6
    - ☐ 15
    - ☐ 0
16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 3
    - ☐ la variable y vaut 5
17. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `int k;`
18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
19. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse lexicale
    - ☐ analyse syntaxique
    - ☐ analyse harmonique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 6
- ☐ 0
- ☐ 15

2. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ main
   - ☐ begin
   - ☐ init
3. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ la division du programme en zones homogènes échoue
4. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ jouer de la musique
5. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ tour à tour, un petit peu à chaque fois
6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3
9. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 4
- ☐ 8

10. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

11. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains

12. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

14. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

15. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ que l'on veut voir tous les avertissements

16. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ ouvrir un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un répertoire

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`

18. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ des processus
- ☐ certaines données de la mémoire de travail

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`

20. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
2. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
3. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!=`
   - ☐ `!`
   - ☐ $\neq$
4. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
5. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int k;`
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail
   - ☐ des processus
   - ☐ en temps d'accès
7. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
8. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`
9. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
10. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
12. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
13. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée

14. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8

15. Un bit est :

- ☐ l'instruction qui met fin à un programme
- ☐ un chiffre binaire (0 ou 1)
- ☐ un battement d'horloge processeur
- ☐ la longueur d'un mot mémoire

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 0

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8

18. Le bus système sert à :

- ☐ Écrire des données sur le dique dur
- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 4
- ☐ 8

20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`

2. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

4. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop

5. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire

6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
   - ☐ 16
   - ☐ 6
   - ☐ 20
   - ☐ 3

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

8. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3

10. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `new TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `kwrite TP4`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

12. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`
- ☐ `1 2 3 1 2`

14. Le langage C est un langage

- ☐ lu, écrit, parlé
- ☐ interprété
- ☐ composé
- ☐ compilé

15. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ include
- ☐ init
- ☐ main
- ☐ begin

16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[5];`
- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`
- ☐ `int tab[] = 5;`
- ☐ `int[] new tableau(5);`

19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 0
- ☐ 15
- ☐ 6

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`

3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`

4. Le bus système sert à :

- ☐ Écrire des données sur le dique dur
- ☐ transporter les processus du tourniquet au processeur
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Arriver à l'heure en cours

5. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ des processus

6. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

7. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran

8. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 1
- ☐ 6
- ☐ 42

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

10. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ changer de répertoire courant
- ☐ ouvrir un fichier texte
- ☐ créer un répertoire

11. Le langage C est un langage

- ☐ compilé
- ☐ interprété
- ☐ composé
- ☐ lu, écrit, parlé

12. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6

13. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int %d;`
- ☐ `int loop n;`
- ☐ `int k;`

14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

18. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier
- ☐ changer de répertoire courant

19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `mkdir TP4`
- ☐ `kwrite TP4`

20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`
- ☐ l'analyse sémantique
- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique
- ☐ l'édition de liens

4. Quels calculs peut-on programmer en programmation structurée ?
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`

6. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
- ☐ un caractère interdit en C
- ☐ une variable non déclarée
- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante

7. L'ordonnancement par tourniquet permet :
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran

8. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = 4`

10. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
- ☐ `mkdir TP4`
- ☐ `new TP4`
- ☐ `kwrite TP4`
- ☐ `yppasswd`

11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7

12. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
13. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ les fichiers du disque
    - ☐ en temps d'accès
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`

15. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale
16. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

17. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante
18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
3. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ composé
   - ☐ compilé
4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
5. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
6. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3
7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 16
   - ☐ 8
9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique
11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ rien

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

12. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
13. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 0
    - ☐ 42
    - ☐ 6
14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
15. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ ne comportent aucune erreur
16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable y vaut 5
17. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
18. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ begin
    - ☐ init
    - ☐ main
19. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ le programme affiche `****`
    - ☐ le programme affiche x
    - ☐ la variable x vaut 16
    - ☐ la variable x vaut $-\frac{1}{2}$

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`

2. L'ordonnancement par tourniquet permet :
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle

3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 4
- ☐ 16
- ☐ 0
- ☐ 8

4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"

5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`

7. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

8. Quel est l'opérateur de différence en C :
   - ☐ `!`
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!=`

9. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire

10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

11. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule en trop
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ une accolade manquante

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0

13. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5

15. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur

16. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ init
    - ☐ main
    - ☐ include

17. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ en temps d'accès
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque

20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 1
- ☐ 6
- ☐ 0
- ☐ 42

2. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`

3. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ `!=`
- ☐ `!`
- ☐ $\neq$

4. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien

5. Le bus système sert à :

- ☐ Arriver à l'heure en cours
- ☐ Écrire des données sur le dique dur
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ transporter les processus du tourniquet au processeur

6. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

7. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ un composant qui contient la liste des fichiers du système

8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

9. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

10. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ répéter un bloc tant qu'une condition est vérifée

11. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`

13. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ main
    - ☐ init
    - ☐ include

14. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1 0

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 16
    - ☐ 4
    - ☐ 0

17. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`

19. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`

20. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ transporter les processus du tourniquet au processeur
2. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `<>`
   - ☐ `!`
   - ☐ `!=`
3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
4. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8
6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ 0 0 1 1 2 2
   - ☐ 0 1 0 1 0 1
   - ☐ 0 1 2 0 1 2
   - ☐ 1 2 3 1 2
8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
9. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
10. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
11. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système

12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0

13. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ les fichiers du disque
    - ☐ en temps d'accès
    - ☐ certaines données de la mémoire de travail
    - ☐ des processus

14. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6

15. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant

16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

17. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements

18. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ un caractère interdit en C

19. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran

20. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
2. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 0`
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = 4`
5. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
7. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`
   - ☐ `** *** **** *****`
   - ☐ `** ** ** ** ** **`
8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
9. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
   - ☐ analyse harmonique
   - ☐ analyse sémantique

10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

11. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ include
    - ☐ init
    - ☐ begin

12. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien

16. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

17. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `loop i;`

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`

19. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique

20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"

3. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien

4. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)

5. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 1 2 3 4 5 6 7

6. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ composé

7. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 16
   - ☐ 3
   - ☐ 6

8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`

9. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de doubler la mémoire disponible

10. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 0
- ☐ 16
- ☐ 8

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
14. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
15. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`
16. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse harmonique
18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`

19. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`
20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ les fichiers du disque

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
2. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
3. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
4. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 4
   - ☐ 8
   - ☐ 0
5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 16
   - ☐ 3
   - ☐ 6
7. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
9. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
11. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 16
    - ☐ le bus explose
    - ☐ le terminal affiche 8
    - ☐ la case mémoire 8 contiendra 0
12. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

14. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale
15. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue
16. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
17. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

19. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
20. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0

3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 5`
   - ☐ `j = 0`

4. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`

7. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 16
   - ☐ 0
   - ☐ 4

8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8
   - ☐ 8 2
   - ☐ 8 6 4 2 0

9. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x

11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define N 3`

12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

13. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 6
- ☐ 10
- ☐ 15

14. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse sémantique

15. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int[] new tableau(5);`
- ☐ `int toto[taille=5];`
- ☐ `int tab[] = 5;`
- ☐ `char tableau[5];`
- ☐ `int toto[5];`

16. Un bit est :

- ☐ l'instruction qui met fin à un programme
- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)
- ☐ la longueur d'un mot mémoire

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

18. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source

19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

2. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti

4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le terminal affiche 8
   - ☐ le bus explose

5. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1`
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2`

6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`

7. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 42
- ☐ 1
- ☐ 6

8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`

☐ `Majeur Mineur`

☐ rien

☐ `Majeur`

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

☐ `Mineur`

☐ `Majeur`

☐ `Majeur Mineur`

☐ rien

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

☐ le programme affiche "Faux"

☐ la variable x vaut 3

☐ la variable y vaut 5

☐ la variable x vaut 5 et la variable y vaut 3

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`

☐ `#include <studio.h>`

☐ `#include <stdio.h>`

☐ `#appart <stdlib.h>`

13. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

☐ `j = 5`

☐ `j = 4`

☐ `j = 0`

☐ `j = %d`

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`

☐ `0 1 2 3 0 1 2`

☐ `0 1 2 0 1 2 3`

☐ `0 0 1 1 2 2 3`

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 1 2 3 4

☐ 4 3 2 1

☐ 0 1 2 3 4

☐ 4 3 2 1 0

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc prog.c -o -Wall prog.exe`

☐ `gcc prog.exe -Wall -o prog.c`

☐ `gcc -Wall prog.c -o prog.exe`

☐ `gcc -Wall prog.exe -o prog.c`

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

☐ la variable y vaut 5

☐ la variable x vaut 5 et la variable y vaut 0

☐ la variable x vaut 0

☐ le programme affiche "Faux"

18. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade manquante

☐ un point-virgule manquant

☐ un point-virgule en trop

☐ une accolade en trop

19. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Majeur`

☐ `Majeur Mineur`

☐ `Mineur`

☐ rien

20. Quel est l'opérateur de différence en C :

☐ `<>`

☐ $\neq$

☐ `!=`

☐ `!`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

2. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

3. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier
- ☐ changer de répertoire courant

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ des processus
- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16

8. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte

9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 0 1 1 2 2 3`

10. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

    ☐ `printf("x=%d et y=%d\n",x,y);`

    ☐ `printf("x=%x et y=%y\n");`

    ☐ `printf("x=%d et y=%d\n,x,y");`

    ☐ `printf("x=%d et y=%d\n",x y);`

11. Le bus système sert à :

    ☐ Écrire des données sur le dique dur

    ☐ transporter les processus du tourniquet au processeur

    ☐ Transférer des données et intructions entre processeur et mémoire

    ☐ Arriver à l'heure en cours

12. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

    ☐ une accolade manquante

    ☐ un point-virgule en trop

    ☐ une accolade en trop

    ☐ un point-virgule manquant

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

    ☐ `#include <studio.h>`

    ☐ `#include <studlib.h>`

    ☐ `#appart <stdlib.h>`

    ☐ `#include <stdio.h>`

14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

    ☐ `for(i=0;i<5;i=i+1)`

    ☐ `for(i=0;i<=5;i=i+1)`

    ☐ `for(i=1;i<=5;i=i+1)`

    ☐ `for(i=1;i<5;i=i+1)`

15. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

    ☐ 0 2 4 6 8

    ☐ 0 2 4 6

    ☐ 0 1 2 3 4 5 6

    ☐ 0 1 2 3 4 5 6 7

16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

    ☐ 4 3 2 1 0

    ☐ 4 3 2 1

    ☐ 0 1 2 3

    ☐ 0 1 2 3 4

17. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :

    ☐ 10

    ☐ 6

    ☐ 0

    ☐ 15

18. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

    ☐ répéter un bloc tant qu'une condition est vérifée

    ☐ mettre les blocs en séquence les uns à la suite des autres

    ☐ retourner un bloc

    ☐ sélectionner entre deux blocs à l'aide d'une condition

19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    ☐ comportent une erreur qui ne sera pas détectée

    ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

    ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

    ☐ ne comportent aucune erreur

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :

    ☐ `Majeur`

    ☐ rien

    ☐ `Majeur Mineur`

    ☐ `Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 8
- ☐ 0

2. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti

3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`

5. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0

7. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`
   - ☐ `int loop n;`

8. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

9. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 6
- ☐ 1
- ☐ 0

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"

11. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
    - ☐ détruire un fichier
    - ☐ changer de répertoire courant
    - ☐ récupérer un programme arrêté avec la commande `ab`
12. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante
13. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ composé
    - ☐ interprété
14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
15. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale
16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?
    - ☐ `j = 0`
    - ☐ `j = 4`
    - ☐ `j = 5`
    - ☐ `j = %d`
18. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
19. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
20. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail

2. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le terminal affiche 8

3. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ composé
   - ☐ lu, écrit, parlé

4. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6

5. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 5`
   - ☐ `j = 0`

8. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 6
   - ☐ 16
   - ☐ 3

9. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée

10. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 4
    - ☐ 16
    - ☐ 0
    - ☐ 8

11. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `loop i;`
12. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
    - ☐ `kwrite TP4`
15. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
16. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
    - ☐ afficher le contenu d'un fichier texte
17. Si cette erreur apparaît à la compilation :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 15
    - ☐ 0
    - ☐ 6
    - ☐ 10
19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur

2. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une gamme de fréquence de fonctionnement du processeur

3. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

5. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 8
   - ☐ 4

6. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
   - ☐ `new TP4`
   - ☐ `mkdir TP4`

7. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"

9. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme

10. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0

11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
```

```
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`

12. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ main
    - ☐ init
    - ☐ include
13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = 4`

15. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `loop i;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
17. Quel est l'opérateur de différence en C :
    - ☐ $\neq$
    - ☐ `!=`
    - ☐ `<>`
    - ☐ `!`
18. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
19. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ $\neq$
   - ☐ `<>`
   - ☐ `!`

2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien

3. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 10
   - ☐ 15
   - ☐ 6
   - ☐ 0

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x

5. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

6. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`

7. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 3
   - ☐ 20
   - ☐ 16
   - ☐ 6

8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3

9. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 16
   - ☐ 4
   - ☐ 8

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`

11. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

12. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
13. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
14. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

18. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 1
- ☐ 42
- ☐ 0

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`

20. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
2. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
4. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ composé
   - ☐ lu, écrit, parlé
5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 42
   - ☐ 6
   - ☐ 0
7. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `1 2 3 1 2`
11. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le bus explose

☐ la case mémoire 8 contiendra 0

☐ le terminal affiche 8

☐ la case mémoire 8 contiendra 16

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

☐ la variable x vaut 3

☐ la variable y vaut 5

☐ la variable x vaut 5 et la variable y vaut 3

☐ le programme affiche "Faux"

13. Un bit est :

☐ la longueur d'un mot mémoire

☐ un chiffre binaire (0 ou 1)

☐ l'instruction qui met fin à un programme

☐ un battement d'horloge processeur

14. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

☐ ne comportent aucune erreur

☐ comportent une erreur qui ne sera pas détectée

☐ comportent une erreur qui sera détectée au cours de l'édition de lien

☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

15. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0

☐ 0 1 2 3 4

☐ 0 1 2 3

☐ 4 3 2 1

16. Dans la commande gcc, l'option `-Wall` signifie :

☐ qu'on veut changer alétoirement de fond d'écran

☐ que l'on veut voir tous les avertissements

☐ qu'il faut lancer un déboggueur

☐ qu'il faut indenter le fichier source

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`

☐ `0 1 2 3 0 1 2`

☐ `0 0 1 1 2 2 3`

☐ `0 1 2 0 1 2 3`

18. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Mineur`

☐ `Majeur Mineur`

☐ `Majeur`

☐ rien

19. Un fichier source est :

☐ un document illisible pour les humains

☐ un fichier texte qui sera traduit en instructions processeur

☐ un document qui doit être protégé

☐ un document de référence du système

☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche "Faux"

☐ le terminal affiche `5`

☐ le terminal affiche `x = 2`

☐ le terminal affiche `x = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire

2. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
   - ☐ `new TP4`

3. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvrir un fichier texte
   - ☐ créer un fichier texte
   - ☐ créer un répertoire

4. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 15
   - ☐ 6
   - ☐ 10
   - ☐ 0

5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ rien

6. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`

8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7

9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui ne sera pas détectée

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0

11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?

☐ `1 2 1 2 3`

☐ `0 1 2 0 1 2`

☐ `0 1 0 1 0 1 0 1`

☐ `0 0 0 1 1 1`

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

☐ `for(i=1;i<5;i=i+1)`

☐ `for(i=0;i<5;i=i+1)`

☐ `for(i=0;i<=5;i=i+1)`

☐ `for(i=1;i<=5;i=i+1)`

13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc -Wall prog.c -o prog.exe`

☐ `gcc -Wall prog.exe -o prog.c`

☐ `gcc prog.exe -Wall -o prog.c`

☐ `gcc prog.c -o -Wall prog.exe`

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `0 1 0 1 0 1`

☐ `1 2 3 1 2`

☐ `0 0 1 1 2 2`

☐ `0 1 2 0 1 2`

15. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade en trop

☐ une accolade manquante

☐ un point-virgule manquant

☐ un point-virgule en trop

16. L'ordonnancement par tourniquet permet :

☐ d'afficher des ronds colorés à l'écran

☐ de ne pas perdre de temps avec la commutation de contexte

☐ de doubler la mémoire disponible

☐ d'entretenir l'illusion que les processus tournent en parallèle

17. Quels calculs peut-on programmer en programmation structurée ?

☐ certains programmes sont de vrais plats de spaghetti

☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

18. Le bus système sert à :

☐ Transférer des données et intructions entre processeur et mémoire

☐ Écrire des données sur le dique dur

☐ Arriver à l'heure en cours

☐ transporter les processus du tourniquet au processeur

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ le programme affiche x

☐ le programme affiche `****`

☐ la variable x vaut $-\frac{1}{2}$

☐ la variable x vaut 16

20. Un programme en langage C doit comporter une et une seule définition de la fonction :

☐ init

☐ include

☐ begin

☐ main

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** −0,5 **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 10
- ☐ 0
- ☐ 15

3. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`

4. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre
- ☐ tous ensemble

5. Un fichier source est :

- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document de référence du système
- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains
- ☐ un fichier texte qui sera traduit en instructions processeur

6. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"

8. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une unité de calcul spécialisée de l'ordinateur

9. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut indenter le fichier source

10. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 16
- ☐ 3
- ☐ 20
- ☐ 6

11. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

12. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
14. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
17. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse syntaxique
    - ☐ analyse harmonique
    - ☐ analyse lexicale
    - ☐ analyse sémantique
18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
19. Si cette erreur apparaît à la compilation :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫ que doit-on chercher dans le programme ?
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 4
    - ☐ 0
    - ☐ 16

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1

2. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

3. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`

5. L'ordonnancement par tourniquet permet :

- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran

6. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ la division du programme en zones homogènes échoue
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"

8. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`

9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`

- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse harmonique

11. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `int %d;`
- ☐ `loop i;`
- ☐ `int k;`

12. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?
- ☐ `1 2 1 2 3`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`

15. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 2

16. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur

17. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`

18. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
- ☐ 3
- ☐ 20
- ☐ 16
- ☐ 6

19. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

2. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`

3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `1 2 1 2 3`

4. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define taille = 3`
- ☐ `#define N 3`

5. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `** *** **** *****`

6. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme

7. Le langage C est un langage

- ☐ lu, écrit, parlé
- ☐ interprété
- ☐ composé
- ☐ compilé

8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = 4`

9. Un bit est :

- ☐ la longueur d'un mot mémoire
- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme

10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`

11. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ la division du programme en zones homogènes échoue
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

12. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ main
- ☐ include
- ☐ init

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

14. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ un point-virgule manquant
- ☐ une accolade manquante

15. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ changer de répertoire courant
- ☐ créer un fichier texte
- ☐ ouvrir un fichier texte
- ☐ créer un répertoire

16. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Arriver à l'heure en cours
- ☐ Écrire des données sur le dique dur

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

19. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

20. Un fichier source est :

- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains
- ☐ un document de référence du système
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 0
☐ 4
☐ 8
☐ 16

2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 16
☐ 4
☐ 8
☐ 0

3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3
☐ 4 3 2 1
☐ 0 1 2 3 4
☐ 4 3 2 1 0

4. Quel est l'opérateur de différence en C :

☐ `!=`
☐ `<>`
☐ `!`
☐ $\neq$

5. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0
☐ 1 2 3 4
☐ 0 1 2 3 4
☐ 4 3 2 1

6. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ un point-virgule en trop
☐ une accolade manquante
☐ un point-virgule manquant
☐ une accolade en trop

7. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

☐ `char tableau[5];`
☐ `int tab[] = 5;`
☐ `int[] new tableau(5);`
☐ `int toto[taille=5];`
☐ `int toto[5];`

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

☐ la variable y vaut 5
☐ le programme affiche "Faux"
☐ la variable x vaut 5 et la variable y vaut 3
☐ la variable x vaut 3

9. Une *segmentation fault* est une erreur qui survient lorsque :

☐ la division du programme en zones homogènes échoue
☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

10. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?

☐ une variable non déclarée
☐ une directive préprocesseur `#include` manquante
☐ un caractère interdit en C
☐ une faute de frappe dans un appel de fonction

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `x = 2`
☐ le terminal affiche `x = 5`
☐ le terminal affiche `5`
☐ le terminal affiche "Faux"

12. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`
13. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse harmonique
14. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 0
    - ☐ le terminal affiche 8
    - ☐ la case mémoire 8 contiendra 16
    - ☐ le bus explose
15. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
16. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ init
    - ☐ include
    - ☐ main
17. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
18. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
19. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 0
    - ☐ 1
    - ☐ 42
    - ☐ 6
20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ changer de répertoire courant
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
2. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`
3. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
4. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 42
   - ☐ 0
   - ☐ 6
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
7. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 0 1 1 2 2 3`
9. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 8
    - ☐ 16
    - ☐ 4
11. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur`

12. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

13. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 0
- ☐ 15
- ☐ 10

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

16. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ begin
- ☐ main
- ☐ include

17. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur

18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

19. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'entretenir l'illusion que les processus tournent en parallèle

20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse lexicale

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"

2. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`

3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :

   - ☐ 4
   - ☐ 0
   - ☐ 16
   - ☐ 8

4. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

   - ☐ `#define N 3`
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`

5. Un registre du processeur est :

   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur

6. Un programme en langage C doit comporter une et une seule définition de la fonction :

   - ☐ include
   - ☐ init
   - ☐ main
   - ☐ begin

7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?

   - ☐ `0 1 0 1 0 1`
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2`

9. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `char tableau[5];`

10. Une *segmentation fault* est une erreur qui survient lorsque :

    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

11. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre

12. Le langage C est un langage

- ☐ interprété
- ☐ lu, écrit, parlé
- ☐ compilé
- ☐ composé

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ retourner un bloc
- ☐ mettre les blocs en séquence les uns à la suite des autres

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`

16. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

18. Un fichier source est :

- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3

3. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 8 2
   - ☐ 0 2 4 6 8

4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2 3`

5. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`

9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`

11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 4
- ☐ 16

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale
    - ☐ analyse sémantique
13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 0
14. Le langage C est un langage
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
15. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ main
    - ☐ begin
    - ☐ include
16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 0
    - ☐ 15
    - ☐ 6
    - ☐ 10
17. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int %d;`
18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
19. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 42
    - ☐ 0
    - ☐ 6
    - ☐ 1
20. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `char tableau[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[5];`
2. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse harmonique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
4. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
5. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 6
   - ☐ 3
   - ☐ 16
7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien
8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Majeur Mineur`
11. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 8 6 4 2 0

☐ 8 6 4 2

☐ 8 2

☐ 0 2 4 6 8

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

☐ 1 2 1 2 3

☐ 0 1 2 0 1 2

☐ 0 1 0 1 0 1 0 1

☐ 0 0 0 1 1 1

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studio.h>`

☐ `#appart <stdlib.h>`

☐ `#include <stdio.h>`

☐ `#include <studlib.h>`

14. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

☐ `printf("x=%d et y=%d\n",x y);`

☐ `printf("x=%d et y=%d\n,x,y");`

☐ `printf("x=%d et y=%d\n",x,y);`

☐ `printf("x=%x et y=%y\n");`

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1

☐ 1 2 3 4

☐ 0 1 2 3 4

☐ 4 3 2 1 0

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `x = 5`

☐ le terminal affiche "Faux"

☐ le terminal affiche `x = 2`

☐ le terminal affiche `5`

17. Sous unix (ou linux), la commande `ls` permet de :

☐ afficher le contenu d'un fichier texte

☐ afficher la liste de fichiers contenus dans un répertoire

☐ compiler un programme

☐ voir des clips musicaux

18. Un bit est :

☐ l'instruction qui met fin à un programme

☐ la longueur d'un mot mémoire

☐ un chiffre binaire (0 ou 1)

☐ un battement d'horloge processeur

19. Quel est l'opérateur de différence en C :

☐ `!=`

☐ `<>`

☐ $\neq$

☐ `!`

20. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ 0 0 1 1 2 2

☐ 1 2 3 1 2

☐ 0 1 2 0 1 2

☐ 0 1 0 1 0 1

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ begin
   - ☐ main
   - ☐ init
2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 16
   - ☐ 4
   - ☐ 8
3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
4. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
5. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 3
   - ☐ 16
   - ☐ 6
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
8. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
   - ☐ une faute de frappe dans un appel de fonction
9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
10. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 8 6 4 2 0
    - ☐ 8 2
    - ☐ 0 2 4 6 8
    - ☐ 8 6 4 2
11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

12. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

15. Quel est l'opérateur de différence en C :
    - ☐ $\neq$
    - ☐ `!`
    - ☐ `!=`
    - ☐ `<>`

16. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
    - ☐ `new TP4`

17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`

18. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

19. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 1
- ☐ 0
- ☐ 42

20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition

2. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte

3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4

4. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`

5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8

6. L'ordonnancement par tourniquet permet :
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de doubler la mémoire disponible
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'afficher des ronds colorés à l'écran

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5

8. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = 4`
   - ☐ `j = %d`

10. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3

11. Le langage C est un langage
    - ☐ compilé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ composé

12. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `<>`
    - ☐ `!`
    - ☐ $\neq$

13. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens

14. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`

16. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6 8

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 0 1 0 1`
    - ☐ `0 0 1 1 2 2`
    - ☐ `1 2 3 1 2`
    - ☐ `0 1 2 0 1 2`

18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `x = 5`

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ en temps d'accès
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ include
   - ☐ main
   - ☐ init
2. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ le programme affiche ****
   - ☐ la variable x vaut $-\frac{1}{2}$
4. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
6. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
7. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
8. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ ** *** **** *****
   - ☐ **** **** **** ****
   - ☐ ***** **** *** ***
   - ☐ ** ** ** ** ** **
9. Un fichier source est :
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un fichier texte qui sera traduit en instructions processeur
10. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 0
    - ☐ 15
    - ☐ 6

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

12. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
13. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
14. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
15. Un registre du processeur est :
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
16. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ les fichiers du disque
    - ☐ des processus
17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`

18. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
19. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ `Mineur`

20. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
   - ☐ une faute de frappe dans un appel de fonction

2. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"

4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"

5. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Écrire des données sur le dique dur

6. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3

8. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n,x,y");`

9. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`

11. Le langage C est un langage
    - ☐ compilé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ composé

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`

13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

14. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ l'instruction qui met fin à un programme
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 1
- ☐ 42
- ☐ 0

16. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`

18. Quel est l'opérateur de différence en C :

- ☐ `!=`
- ☐ $\neq$
- ☐ `<>`
- ☐ `!`

19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 8
- ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
2. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ init
   - ☐ begin
   - ☐ include
3. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`
   - ☐ `***** **** *** ***`
4. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
6. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8
   - ☐ 8 2
   - ☐ 8 6 4 2
7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
8. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
9. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int k;`
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `loop i;`
10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
11. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse sémantique
    - ☐ analyse syntaxique
    - ☐ analyse harmonique

13. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

14. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

16. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble

18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`

19. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ ouvrir un fichier texte
- ☐ créer un répertoire
- ☐ changer de répertoire courant

20. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée
- ☐ un caractère interdit en C

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti

2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 4
   - ☐ 8

3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition

4. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`

5. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`

6. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8

7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2 3`

9. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `<>`
   - ☐ `!=`

10. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1

11. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système

12. Un bit est :

- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme
- ☐ la longueur d'un mot mémoire
- ☐ un battement d'horloge processeur

13. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 42
- ☐ 1
- ☐ 6

14. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse sémantique
- ☐ l'analyse des entrées clavier
- ☐ l'édition de liens
- ☐ l'analyse harmonique

15. Un registre du processeur est :

- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

16. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme
- ☐ voir des clips musicaux

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"

18. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?

- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée
- ☐ une faute de frappe dans un appel de fonction
- ☐ un caractère interdit en C

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements

20. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0

2. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)

3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

4. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`

5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"

6. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0

7. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0

8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

11. Un fichier source est :
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
12. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
13. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 0
- ☐ 1
- ☐ 6

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
16. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop
    - ☐ une accolade en trop
17. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ le programme affiche x

20. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `loop i;`
    - ☐ `int %d;`
    - ☐ `int loop n;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0

3. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur
- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours

4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

5. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.c -o -Wall prog.exe`

6. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

7. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

8. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte

9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`

- ☐ l'analyse harmonique
- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"

12. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
13. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
14. Quel est l'opérateur de différence en C :
    - ☐ $\neq$
    - ☐ `!`
    - ☐ `<>`
    - ☐ `!=`
15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
16. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
    - ☐ créer un répertoire
17. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ les fichiers du disque
    - ☐ en temps d'accès
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
18. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 4
    - ☐ 16
    - ☐ 0
19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
20. Le langage C est un langage
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ composé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = %d`

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"

4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 4`

6. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse sémantique
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
   - ☐ analyse harmonique

7. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
   - ☐ `mkdir TP4`

8. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche x

10. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible

11. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

12. Le langage C est un langage
    - ☐ composé
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ interprété
13. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `int k;`
    - ☐ `loop i;`
14. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ ouvrir un fichier texte
15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
16. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque
    - ☐ des processus
    - ☐ en temps d'accès
17. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
18. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`

20. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
2. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
3. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
5. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
9. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
10. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
11. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4

12. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Écrire des données sur le dique dur
- ☐ Transférer des données et intructions entre processeur et mémoire

13. L'ordonnancement par tourniquet permet :

- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5

15. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`

16. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`

17. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse harmonique
- ☐ analyse lexicale
- ☐ analyse sémantique
- ☐ analyse syntaxique

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = %d`

19. Un fichier source est :

- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document de référence du système
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

20. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ `!=`
- ☐ $\neq$
- ☐ `!`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse sémantique
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
2. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ init
   - ☐ include
   - ☐ begin
3. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ changer de répertoire courant
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le terminal affiche 8
5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
6. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`
8. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
10. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2 0
    - ☐ 8 6 4 2
11. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
12. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

☐ la variable x vaut 5 et la variable y vaut 0
☐ la variable y vaut 5
☐ le programme affiche "Faux"
☐ la variable x vaut 0

14. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
☐ 16
☐ 3
☐ 6
☐ 20

15. Sous unix (ou linux), la commande `ls` permet de :
☐ voir des clips musicaux
☐ afficher le contenu d'un fichier texte
☐ afficher la liste de fichiers contenus dans un répertoire
☐ compiler un programme

16. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
☐ `yppasswd`
☐ `kwrite TP4`
☐ `new TP4`
☐ `mkdir TP4`

17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
☐ 0 1 2 3 4
☐ 0 1 2 3
☐ 4 3 2 1
☐ 4 3 2 1 0

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?
☐ 0 1 0 1 0 1 0 1
☐ 0 0 0 1 1 1
☐ 0 1 2 0 1 2
☐ 1 2 1 2 3

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?
☐ 0 1 2 0 1 2
☐ 0 0 1 1 2 2 3
☐ 0 1 2 3 0 1 2
☐ 0 1 2 0 1 2 3

20. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
☐ 0 2 4 6 8
☐ 0 2 4 6
☐ 0 1 2 3 4 5 6 7
☐ 0 1 2 3 4 5 6

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0

2. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8

3. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
   - ☐ `int tab[] = 5;`

4. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `***** **** *** ***`
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`

5. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

7. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `!=`
   - ☐ `<>`

8. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 0`
   - ☐ `j = 4`

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 0 1 1 1`
    - ☐ `0 1 0 1 0 1 0 1`
    - ☐ `1 2 1 2 3`
    - ☐ `0 1 2 0 1 2`

11. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ certaines données de la mémoire de travail

12. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16

15. Le langage C est un langage

- ☐ composé
- ☐ interprété
- ☐ compilé
- ☐ lu, écrit, parlé

16. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ changer de répertoire courant

17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 10
- ☐ 15
- ☐ 0

19. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

2. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 0
- ☐ 42
- ☐ 1

3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6

4. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`

5. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ en temps d'accès
- ☐ des processus

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ le programme affiche x
- ☐ la variable x vaut 16

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

8. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int %d;`
- ☐ `int k;`
- ☐ `loop i;`
- ☐ `int loop n;`

11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`

13. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse sémantique
- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse lexicale

14. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition

15. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`

16. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte

17. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `kwrite TP4`

18. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

20. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ changer de répertoire courant

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5

2. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?

   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
   - ☐ des processus

5. Sur unix (ou linux), la commande `mkdir` permet de :

   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire

6. Une *segmentation fault* est une erreur qui survient lorsque :

   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5

8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?

   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`

10. Sous unix (ou linux), la commande `ls` permet de :

    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme

11. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur

12. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 0
    - ☐ la case mémoire 8 contiendra 16
    - ☐ le terminal affiche 8
    - ☐ le bus explose

14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 0`
    - ☐ `j = 4`
    - ☐ `j = 5`

16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur Mineur`

17. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`

19. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire

20. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique
    - ☐ détruire un fichier
    - ☐ changer de répertoire courant

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 8
- ☐ 0

2. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte
3. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès
   - ☐ des processus
4. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 0
- ☐ 16

5. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
6. Un fichier source est :
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
7. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
8. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
9. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 16
- ☐ 3
- ☐ 20

10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`

11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`

12. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran

13. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur

14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int %d;`
- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `loop i;`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

17. Le langage C est un langage

- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ composé
- ☐ interprété

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

20. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le langage C est un langage
   - ☐ compilé
   - ☐ composé
   - ☐ interprété
   - ☐ lu, écrit, parlé
2. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`
3. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
5. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
6. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
8. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ init
   - ☐ include
   - ☐ begin
9. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 6
   - ☐ 42
   - ☐ 0
10. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ la division du programme en zones homogènes échoue
11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16

16. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ des processus

17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

18. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`

- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique
- ☐ l'édition de liens
- ☐ l'analyse sémantique

19. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ un composant qui contient la liste des fichiers du système

20. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
2. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
3. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ interprété
   - ☐ composé
4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 16
5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 16
   - ☐ 8
   - ☐ 4
7. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
8. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`
   - ☐ $\neq$
9. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ include
    - ☐ begin
    - ☐ init
11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ rien

12. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)

13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 4`
    - ☐ `j = %d`
    - ☐ `j = 5`
    - ☐ `j = 0`

15. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 20
    - ☐ 16
    - ☐ 3
    - ☐ 6

16. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`

18. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier

19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 16
    - ☐ 4
    - ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2 0

2. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`

4. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`

5. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ `!=`
- ☐ `!`
- ☐ $\neq$

6. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = %d`

7. Un registre du processeur est :

- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`

10. Le langage C est un langage

- ☐ lu, écrit, parlé
- ☐ composé
- ☐ compilé
- ☐ interprété

11. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse harmonique
- ☐ analyse syntaxique
- ☐ analyse lexicale
- ☐ analyse sémantique

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

14. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

15. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ la division du programme en zones homogènes échoue
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

16. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ transporter les processus du tourniquet au processeur
- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours

17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

18. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x,y);`

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 4
- ☐ 16

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 6
- ☐ 42
- ☐ 1

2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

3. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Transférer des données et intructions entre processeur et mémoire

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5

5. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`

6. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système

7. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`

8. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
   - ☐ créer un fichier texte
   - ☐ changer de répertoire courant

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`

10. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti

11. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3

12. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`

13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

14. Un bit est :

- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme
- ☐ un battement d'horloge processeur
- ☐ la longueur d'un mot mémoire

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`

16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

17. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une directive préprocesseur `#include` manquante
- ☐ une faute de frappe dans un appel de fonction
- ☐ une variable non déclarée
- ☐ un caractère interdit en C

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ que l'on veut voir tous les avertissements

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 5`

2. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3

4. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 3
   - ☐ 20
   - ☐ 16

5. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`

7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`

8. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

9. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 0
   - ☐ 1
   - ☐ 42
   - ☐ 6

10. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
    - ☐ `int toto[taille=5];`

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
13. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
14. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
15. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

16. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

18. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
2. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
3. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
4. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)
5. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 10
   - ☐ 15
   - ☐ 0
6. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
8. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
9. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
10. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 42
    - ☐ 6
    - ☐ 1
    - ☐ 0
11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6
12. Le langage C est un langage
    - ☐ interprété
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ composé

13. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

14. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ begin
- ☐ include
- ☐ main

15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ retourner un bloc

16. Sous unix (ou linux), la commande `cd` permet de :

- ☐ détruire un fichier
- ☐ changer de répertoire courant
- ☐ ouvir un bureau partagé (common desktop)
- ☐ jouer de la musique
- ☐ récupérer un programme arrêté avec la commande `ab`

17. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 4
- ☐ 0

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`

20. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

2. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
4. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
5. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!`
6. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
7. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers` « `prinft` »
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
10. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
11. Le langage C est un langage
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
12. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ les fichiers du disque
13. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`

16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 0
- ☐ 8

18. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8

2. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8

3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x y);`

4. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de ne pas perdre de temps avec la commutation de contexte

5. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse syntaxique
- ☐ analyse sémantique
- ☐ analyse harmonique

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ les fichiers du disque
- ☐ des processus
- ☐ en temps d'accès
- ☐ certaines données de la mémoire de travail

7. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ répéter un bloc tant qu'une condition est vérifée

8. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`

9. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`

11. Quel est l'opérateur de différence en C :

- ☐ $\neq$
- ☐ `<>`
- ☐ `!`
- ☐ `!=`

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

14. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 4
- ☐ 8

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 0

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

17. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une variable non déclarée
- ☐ un caractère interdit en C
- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante

18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 2

2. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

3. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)

4. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

5. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule en trop
   - ☐ une accolade en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant

6. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ la division du programme en zones homogènes échoue

7. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements

8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 5`
   - ☐ `j = 0`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3

10. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document illisible pour les humains

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
12. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
13. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
14. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
15. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
16. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 5`
    - ☐ `j = 4`
    - ☐ `j = %d`
    - ☐ `j = 0`
17. Quel est l'opérateur de différence en C :
    - ☐ $\neq$
    - ☐ `!`
    - ☐ `!=`
    - ☐ `<>`
18. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
    somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 0
    - ☐ 1
    - ☐ 6
    - ☐ 42
19. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
20. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ main
   - ☐ init
   - ☐ begin
2. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[5];`
   - ☐ `char tableau[5];`
3. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
5. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 16
   - ☐ 8
   - ☐ 4
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 4`
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
11. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

☐ 8 6 4 2

☐ 8 6 4 2 0

☐ 8 2

☐ 0 2 4 6 8

12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

☐ le terminal affiche 8

☐ la case mémoire 8 contiendra 16

☐ la case mémoire 8 contiendra 0

☐ le bus explose

13. Le bus système sert à :

☐ transporter les processus du tourniquet au processeur

☐ Arriver à l'heure en cours

☐ Écrire des données sur le dique dur

☐ Transférer des données et intructions entre processeur et mémoire

14. Sous unix (ou linux), la commande `cd` permet de :

☐ détruire un fichier

☐ ouvir un bureau partagé (common desktop)

☐ récupérer un programme arrêté avec la commande `ab`

☐ jouer de la musique

☐ changer de répertoire courant

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1

☐ 1 2 3 4

☐ 0 1 2 3 4

☐ 4 3 2 1 0

16. Le langage C est un langage

☐ interprété

☐ lu, écrit, parlé

☐ composé

☐ compilé

17. Sur unix (ou linux), la commande `mkdir` permet de :

☐ changer de répertoire courant

☐ créer un fichier texte

☐ ouvrir un fichier texte

☐ créer un répertoire

18. Quels calculs peut-on programmer en programmation structurée ?

☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

☐ certains programmes sont de vrais plats de spaghetti

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `5`

☐ le terminal affiche `x = 2`

☐ le terminal affiche `x = 5`

☐ le terminal affiche "Faux"

20. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

☐ 6

☐ 1

☐ 0

☐ 42

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
2. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
5. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
7. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse harmonique
   - ☐ analyse sémantique
8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`
10. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `new TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `kwrite TP4`
11. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant
    - ☐ jouer de la musique
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
12. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
    - ☐ compiler un programme
13. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

14. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

15. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop
16. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ Transférer des données et intructions entre processeur et mémoire
17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
18. Le langage C est un langage
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ interprété
19. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »`
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
20. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `ls` permet de :

   ☐ afficher le contenu d'un fichier texte

   ☐ compiler un programme

   ☐ afficher la liste de fichiers contenus dans un répertoire

   ☐ voir des clips musicaux

2. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   ☐ 0 1 2 3 4

   ☐ 4 3 2 1 0

   ☐ 4 3 2 1

   ☐ 0 1 2 3

3. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :

   ☐ 6

   ☐ 16

   ☐ 3

   ☐ 20

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :

   ☐ `Majeur Mineur`

   ☐ `Majeur`

   ☐ rien

   ☐ `Mineur`

5. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

   ☐ `loop i;`

   ☐ `int loop n;`

   ☐ `int k;`

   ☐ `int %d;`

6. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?

   ☐ `***** **** *** ***`

   ☐ `** *** **** *****`

   ☐ `** ** ** ** ** **`

   ☐ `**** **** **** ****`

7. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   ☐ le terminal affiche 8

   ☐ la case mémoire 8 contiendra 16

   ☐ le bus explose

   ☐ la case mémoire 8 contiendra 0

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#include <studlib.h>`

   ☐ `#include <stdio.h>`

   ☐ `#appart <stdlib.h>`

   ☐ `#include <studio.h>`

9. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   ☐ `Majeur Mineur`

   ☐ `Mineur`

   ☐ rien

   ☐ `Majeur`

10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

12. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé
    - ☐ interprété
13. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
15. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'on veut changer alétoirement de fond d'écran
16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

17. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6

18. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 4
- ☐ 16

19. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
20. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier

2. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3

3. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
   - ☐ analyse syntaxique

4. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`

5. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains

6. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur

7. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ include
   - ☐ init
   - ☐ begin

8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`
   - ☐ `int loop n;`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5

10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire

11. Quel est l'opérateur de différence en C :
    - ☐ `!`
    - ☐ `<>`
    - ☐ $\neq$
    - ☐ `!=`

12. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2
    - ☐ 8 6 4 2 0

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 3 0 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 0 1 2 3`

14. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
15. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 6
    - ☐ 0
    - ☐ 15
17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0
18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `1 2 3 1 2`
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1`
19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 1 2 3 4
☐ 4 3 2 1 0
☐ 4 3 2 1
☐ 0 1 2 3 4

2. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define taille = N`
☐ `#define N 3`
☐ `#define N =  3`
☐ `#define taille = 3`

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4
☐ 4 3 2 1 0
☐ 4 3 2 1
☐ 0 1 2 3

4. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

☐ `new TP4`
☐ `yppasswd`
☐ `mkdir TP4`
☐ `kwrite TP4`

5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ rien
☐ `Mineur`
☐ `Majeur Mineur`
☐ `Majeur`

6. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` ≪ `prinft` ≫ que doit-on chercher dans le programme ?

☐ une faute de frappe dans un appel de fonction
☐ une variable non déclarée
☐ une directive préprocesseur `#include` manquante
☐ un caractère interdit en C

7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1
☐ 0 1 2 3 4
☐ 4 3 2 1 0
☐ 1 2 3 4

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `0 1 0 1 0 1`
☐ `0 1 2 0 1 2`
☐ `0 0 1 1 2 2`
☐ `1 2 3 1 2`

9. Le langage C est un langage

☐ composé
☐ lu, écrit, parlé
☐ compilé
☐ interprété

10. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ tour à tour, un petit peu à chaque fois
☐ en parallèle, chacun dans un registre
☐ chacun son tour, après que le processus précédent a terminé
☐ tous ensemble

11. Un bit est :

☐ un battement d'horloge processeur
☐ un chiffre binaire (0 ou 1)
☐ la longueur d'un mot mémoire
☐ l'instruction qui met fin à un programme

12. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
13. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse harmonique
14. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2 0
    - ☐ 8 6 4 2
15. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ ne comportent aucune erreur
16. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
17. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 20
    - ☐ 16
    - ☐ 6
    - ☐ 3
18. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ init
    - ☐ main
    - ☐ begin
19. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut 16
    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche `****`
    - ☐ le programme affiche x

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 20
☐ 3
☐ 6
☐ 16

2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 6
☐ 15
☐ 10
☐ 0

3. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc prog.c -o -Wall prog.exe`
☐ `gcc prog.exe -Wall -o prog.c`
☐ `gcc -Wall prog.exe -o prog.c`
☐ `gcc -Wall prog.c -o prog.exe`

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ les fichiers du disque
☐ des processus
☐ en temps d'accès
☐ certaines données de la mémoire de travail

5. Sous unix (ou linux), la commande `cd` permet de :

☐ jouer de la musique
☐ détruire un fichier
☐ ouvir un bureau partagé (common desktop)
☐ changer de répertoire courant
☐ récupérer un programme arrêté avec la commande `ab`

6. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `5`
☐ le terminal affiche `x = 5`
☐ le terminal affiche "Faux"
☐ le terminal affiche `x = 2`

7. Sous unix (ou linux), la commande `ls` permet de :

☐ afficher le contenu d'un fichier texte
☐ afficher la liste de fichiers contenus dans un répertoire
☐ voir des clips musicaux
☐ compiler un programme

8. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0
☐ 1 2 3 4
☐ 0 1 2 3 4
☐ 4 3 2 1

9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ chacun son tour, après que le processus précédent a terminé
☐ tous ensemble
☐ en parallèle, chacun dans un registre
☐ tour à tour, un petit peu à chaque fois

10. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`
☐ `#include <studio.h>`
☐ `#appart <stdlib.h>`
☐ `#include <stdio.h>`

11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define taille = N`
☐ `#define N 3`
☐ `#define taille = 3`
☐ `#define N =  3`

12. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = 5`

15. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un document de référence du système
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"

18. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

2. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ init
   - ☐ main
   - ☐ begin

3. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16

4. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ jouer de la musique

5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 2
- ☐ 8 6 4 2 0

6. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"

7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

8. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

10. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4 5 6

☐ 0 1 2 3 4 5 6 7

☐ 0 2 4 6 8

☐ 0 2 4 6

11. Le langage C est un langage

☐ interprété

☐ compilé

☐ lu, écrit, parlé

☐ composé

12. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 3

☐ 16

☐ 20

☐ 6

13. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

☐ `** *** **** *****`

☐ `***** **** *** ***`

☐ `** ** ** ** ** **`

☐ `**** **** **** ****`

14. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 15

☐ 10

☐ 6

☐ 0

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ rien

☐ `Mineur`

☐ `Majeur Mineur`

☐ `Majeur`

16. Un registre du processeur est :

☐ une gamme de fréquence de fonctionnement du processeur

☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

☐ un composant qui contient la liste des fichiers du système

☐ une unité de calcul spécialisée de l'ordinateur

17. L'ordonnancement par tourniquet permet :

☐ de doubler la mémoire disponible

☐ de ne pas perdre de temps avec la commutation de contexte

☐ d'entretenir l'illusion que les processus tournent en parallèle

☐ d'afficher des ronds colorés à l'écran

18. Sous unix (ou linux), la commande `ls` permet de :

☐ afficher la liste de fichiers contenus dans un répertoire

☐ voir des clips musicaux

☐ compiler un programme

☐ afficher le contenu d'un fichier texte

19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ en temps d'accès

☐ les fichiers du disque

☐ des processus

☐ certaines données de la mémoire de travail

20. Un bit est :

☐ un chiffre binaire (0 ou 1)

☐ la longueur d'un mot mémoire

☐ un battement d'horloge processeur

☐ l'instruction qui met fin à un programme

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`

2. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains

3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `int k;`
   - ☐ `loop i;`

5. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`

6. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
   - ☐ 8
   - ☐ 4
   - ☐ 0
   - ☐ 16

7. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale

8. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
   - ☐ 4
   - ☐ 16
   - ☐ 0
   - ☐ 8

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = 5`

10. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant

11. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)

12. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
15. Le langage C est un langage
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
16. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`
17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé
18. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ $\neq$
    - ☐ `!`
    - ☐ `<>`
19. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#appart <stdlib.h>`

   ☐ `#include <studlib.h>`

   ☐ `#include <stdio.h>`

   ☐ `#include <studio.h>`

2. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?

   ☐ `0 0 1 1 2 2 3`

   ☐ `0 1 2 0 1 2 3`

   ☐ `0 1 2 0 1 2`

   ☐ `0 1 2 3 0 1 2`

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   ☐ le programme affiche "Faux"

   ☐ la variable x vaut 3

   ☐ la variable x vaut 5 et la variable y vaut 3

   ☐ la variable y vaut 5

4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?

   ☐ `0 0 1 1 2 2`

   ☐ `1 2 3 1 2`

   ☐ `0 1 0 1 0 1`

   ☐ `0 1 2 0 1 2`

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   ☐ la variable x vaut 0

   ☐ le programme affiche "Faux"

   ☐ la variable y vaut 5

   ☐ la variable x vaut 5 et la variable y vaut 0

6. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   ☐ la case mémoire 8 contiendra 0

   ☐ le terminal affiche 8

   ☐ la case mémoire 8 contiendra 16

   ☐ le bus explose

7. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

   ☐ `yppasswd`

   ☐ `kwrite TP4`

   ☐ `new TP4`

   ☐ `mkdir TP4`

8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   ☐ rien

   ☐ `Majeur Mineur`

   ☐ `Majeur`

   ☐ `Mineur`

9. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

   ☐ `printf("x=%d et y=%d\n",x y);`

   ☐ `printf("x=%x et y=%y\n");`

   ☐ `printf("x=%d et y=%d\n,x,y");`

   ☐ `printf("x=%d et y=%d\n",x,y);`

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien

11. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
13. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

14. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur
15. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
16. Un fichier source est :
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

18. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
19. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
2. L'ordonnancement par tourniquet permet :
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de doubler la mémoire disponible
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`
4. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 42
   - ☐ 0
   - ☐ 1
   - ☐ 6
6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
7. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse harmonique
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le terminal affiche 8
9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
10. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
11. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`

12. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6

13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%x et y=%y\n");`

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`
- ☐ `new TP4`

15. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ changer de répertoire courant
- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3

20. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 0
- ☐ 6
- ☐ 10

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `new TP4`
   - ☐ `yppasswd`
   - ☐ `mkdir TP4`
2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
4. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
   - ☐ analyse harmonique
5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 42
   - ☐ 0
   - ☐ 6
6. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
7. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ begin
   - ☐ init
   - ☐ main
8. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 15
   - ☐ 6
   - ☐ 10
   - ☐ 0
9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `1 2 3 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `0 0 1 1 2 2`
11. Quel est l'opérateur de différence en C :
    - ☐ `!`
    - ☐ $\neq$
    - ☐ `!=`
    - ☐ `<>`
12. Le langage C est un langage
    - ☐ composé
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ interprété

13. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3

15. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `5`

17. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ jouer de la musique
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant

18. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte

19. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme

20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`
- ☐ `1 2 1 2 3`

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"

3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define N 3`

4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`
- ☐ `1 2 3 1 2`

5. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int %d;`

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ les fichiers du disque
- ☐ en temps d'accès

7. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 0
- ☐ 8
- ☐ 16

8. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti

9. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6

10. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de ne pas perdre de temps avec la commutation de contexte

11. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse syntaxique
- ☐ analyse lexicale

12. Un bit est :

- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ un battement d'horloge processeur
- ☐ l'instruction qui met fin à un programme

13. Le langage C est un langage

- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ interprété
- ☐ composé

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

15. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur

16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose

17. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte

18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

19. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int tab[] = 5;`
- ☐ `int toto[5];`
- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`
- ☐ `int[] new tableau(5);`

20. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 15
- ☐ 10
- ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ jouer de la musique
2. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `<>`
   - ☐ `!=`
   - ☐ `!`
3. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
5. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
6. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
7. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
8. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
9. Si cette erreur apparaît à la compilation :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
10. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
11. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le terminal affiche 8
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 16
12. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`

13. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

14. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 4
- ☐ 16

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

16. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`
- ☐ `int[] new tableau(5);`
- ☐ `int tab[] = 5;`
- ☐ `int toto[5];`

19. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 20
- ☐ 3
- ☐ 6
- ☐ 16

20. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un répertoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x

2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 15
   - ☐ 0
   - ☐ 10

3. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès
   - ☐ des processus

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`

6. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique
   - ☐ l'analyse des entrées clavier
   - ☐ l'édition de liens

7. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
   - ☐ `int[] new tableau(5);`

8. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
   - ☐ analyse harmonique
   - ☐ analyse sémantique

9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
   - ☐ `mkdir TP4`
   - ☐ `new TP4`

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0

11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 1 2 3 4 5 6 7

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
```

```
15
16        ...
17    }
```

- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ 0 1 0 1 0 1
- ☐ 0 1 2 0 1 2
- ☐ 0 0 1 1 2 2
- ☐ 1 2 3 1 2

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

15. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 0
- ☐ 16

16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 0 2 4 6 8

18. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre

19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée

20. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int %d;`
- ☐ `loop i;`
- ☐ `int loop n;`
- ☐ `int k;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `!=`
2. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x
4. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de doubler la mémoire disponible
5. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
7. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`
   - ☐ `***** **** *** ***`
   - ☐ `**** **** **** ****`
8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 8 2
9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2`
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
```

```
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`

11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
12. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
14. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé
15. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 0
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 16
    - ☐ le terminal affiche 8
17. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Mineur`
18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6 7
19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 8
    - ☐ 4
    - ☐ 0
20. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

3. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements

4. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`

5. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`

6. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0

7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4

8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`

10. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur

11. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0

12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `0 1 0 1 0 1`

☐ `0 0 1 1 2 2`

☐ `0 1 2 0 1 2`

☐ `1 2 3 1 2`

13. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 0 1 0 1 0 1`

☐ `0 0 0 1 1 1`

☐ `0 1 2 0 1 2`

☐ `1 2 1 2 3`

14. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 8

☐ 16

☐ 0

☐ 4

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4

☐ 1 2 3 4

☐ 4 3 2 1

☐ 4 3 2 1 0

16. Un bit est :

☐ un chiffre binaire (0 ou 1)

☐ la longueur d'un mot mémoire

☐ un battement d'horloge processeur

☐ l'instruction qui met fin à un programme

17. Quel est l'opérateur de différence en C :

☐ $\neq$

☐ `<>`

☐ `!=`

☐ `!`

18. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ tous ensemble

☐ tour à tour, un petit peu à chaque fois

☐ en parallèle, chacun dans un registre

☐ chacun son tour, après que le processus précédent a terminé

19. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

☐ `Majeur`

☐ `Majeur Mineur`

☐ `Mineur`

☐ rien

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ en temps d'accès

☐ des processus

☐ les fichiers du disque

☐ certaines données de la mémoire de travail

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

2. Un fichier source est :

- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document qui doit être protégé
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document illisible pour les humains

3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6 8
- ☐ 0 2 4 6

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ certaines données de la mémoire de travail

5. Sous unix (ou linux), la commande `cd` permet de :

- ☐ jouer de la musique
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ changer de répertoire courant
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier

6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

7. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur

8. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 3
- ☐ 20
- ☐ 16

9. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

10. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 6
- ☐ 0
- ☐ 1

11. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5

13. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 4
- ☐ 0

14. L'ordonnancement par tourniquet permet :

- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ de doubler la mémoire disponible
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran

15. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

17. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `int %d;`
- ☐ `int k;`
- ☐ `loop i;`

18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

19. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien

20. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

2. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 1 2 3 4

3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`

4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"

5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

6. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16

- ☐ le programme affiche ****
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x

10. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur
11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 0
    - ☐ 16
    - ☐ 4
12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?
    - ☐ `j = 5`
    - ☐ `j = %d`
    - ☐ `j = 0`
    - ☐ `j = 4`
13. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ main
    - ☐ include
    - ☐ begin
14. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
15. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 6
    - ☐ 10
    - ☐ 0
    - ☐ 15
16. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran
17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"
18. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant
19. Le langage C est un langage
    - ☐ composé
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ compilé
20. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
2. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
4. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)
5. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
6. Un fichier source est :
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
7. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 8
   - ☐ 0
   - ☐ 16
8. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 0
   - ☐ 8
   - ☐ 16
9. Le langage C est un langage
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ composé
10. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
11. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`

13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
14. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
15. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
16. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

    qu'est ce qui sera affiché ?
    - ☐ `**** **** **** ****`
    - ☐ `** ** ** ** ** **`
    - ☐ `** *** **** *****`
    - ☐ `***** **** *** ***`
17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
18. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ $\neq$
    - ☐ `!`
    - ☐ `<>`
19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut 16
    - ☐ le programme affiche `****`
    - ☐ le programme affiche x
    - ☐ la variable x vaut $-\frac{1}{2}$
20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `1 2 1 2 3`
    - ☐ `0 1 0 1 0 1 0 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 0 1 1 1`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

2. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
3. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
4. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"

6. Le langage C est un langage
   - ☐ compilé
   - ☐ composé
   - ☐ interprété
   - ☐ lu, écrit, parlé
7. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
8. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

9. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`
- ☐ `1 2 1 2 3`

11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

12. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 1
- ☐ 42
- ☐ 6

13. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

14. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé

15. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`

16. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `!`
    - ☐ `<>`
    - ☐ $\neq$

17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier

18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`

19. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int %d;`
    - ☐ `int k;`

20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8

2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`

3. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`

4. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ changer de répertoire courant
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier

5. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran

6. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur

7. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire

8. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade manquante
- ☐ un point-virgule manquant
- ☐ une accolade en trop

9. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`

10. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur

11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`

12. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `1 2 3 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`

14. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ main
- ☐ include
- ☐ begin

15. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

17. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

18. Un fichier source est :

- ☐ un document de référence du système
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé

19. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse harmonique
- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique
- ☐ l'édition de liens

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ des processus

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1
☐ 0 1 2 3 4
☐ 4 3 2 1 0
☐ 0 1 2 3

2. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 1 2 3 4
☐ 4 3 2 1 0
☐ 0 1 2 3 4
☐ 4 3 2 1

3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define N 3`
☐ `#define taille = N`
☐ `#define N =  3`
☐ `#define taille = 3`

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ le programme affiche x
☐ la variable x vaut $-\frac{1}{2}$
☐ la variable x vaut 16
☐ le programme affiche ****

5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `x = 2`
☐ le terminal affiche `5`
☐ le terminal affiche "Faux"
☐ le terminal affiche `x = 5`

6. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

☐ `int toto[5];`
☐ `int toto[taille=5];`
☐ `int tab[] = 5;`
☐ `char tableau[5];`
☐ `int[] new tableau(5);`

7. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade manquante
☐ un point-virgule en trop
☐ une accolade en trop
☐ un point-virgule manquant

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`
☐ `0 1 2 3 0 1 2`
☐ `0 0 1 1 2 2 3`
☐ `0 1 2 0 1 2 3`

9. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

☐ une directive préprocesseur `#include` manquante
☐ un caractère interdit en C
☐ une faute de frappe dans un appel de fonction
☐ une variable non déclarée

10. Un bit est :

☐ l'instruction qui met fin à un programme
☐ la longueur d'un mot mémoire
☐ un battement d'horloge processeur
☐ un chiffre binaire (0 ou 1)

11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

13. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

14. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

16. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

17. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ 0 0 0 1 1 1
- ☐ 0 1 2 0 1 2
- ☐ 0 1 0 1 0 1 0 1
- ☐ 1 2 1 2 3

19. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'afficher des ronds colorés à l'écran

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 0

☐ 16

☐ 8

☐ 4

2. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

☐ comportent une erreur qui ne sera pas détectée

☐ comportent une erreur qui sera détectée au cours de l'édition de lien

☐ ne comportent aucune erreur

3. Quel est l'opérateur de différence en C :

☐ `<>`

☐ `!=`

☐ $\neq$

☐ `!`

4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

☐ la case mémoire 8 contiendra 0

☐ le bus explose

☐ la case mémoire 8 contiendra 16

☐ le terminal affiche 8

5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ rien

☐ `Majeur`

☐ `Mineur`

☐ `Majeur Mineur`

6. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define N 3`

☐ `#define taille = N`

☐ `#define N =  3`

☐ `#define taille = 3`

7. Dans la commande gcc, l'option `-Wall` signifie :

☐ qu'il faut lancer un déboggueur

☐ qu'on veut changer alétoirement de fond d'écran

☐ qu'il faut indenter le fichier source

☐ que l'on veut voir tous les avertissements

8. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4

☐ 0 1 2 3

☐ 4 3 2 1 0

☐ 4 3 2 1

9. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade en trop

☐ un point-virgule en trop

☐ un point-virgule manquant

☐ une accolade manquante

10. Sur unix (ou linux), la commande `mkdir` permet de :

☐ changer de répertoire courant

☐ ouvrir un fichier texte

☐ créer un répertoire

☐ créer un fichier texte

11. Un registre du processeur est :

☐ une gamme de fréquence de fonctionnement du processeur

☐ un composant qui contient la liste des fichiers du système

☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

☐ une unité de calcul spécialisée de l'ordinateur

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`

13. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 4
- ☐ 0

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 0
- ☐ 6
- ☐ 1

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int %d;`

17. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

18. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`

19. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`

20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse sémantique
- ☐ analyse harmonique
- ☐ analyse syntaxique
- ☐ analyse lexicale

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 points par réponse juste (unique) ; −0,5 points par réponse fausse. Durée : 20 minutes.

1. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
2. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
   - ☐ les fichiers du disque
   - ☐ des processus
3. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
4. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
5. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
   - ☐ ouvrir un fichier texte
6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
9. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ main
    - ☐ init
    - ☐ include
11. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
12. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une variable non déclarée
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
13. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »`
    - ☐ l'édition de liens
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique

14. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 3
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable y vaut 5

16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée

17. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`

18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 15
    - ☐ 0
    - ☐ 6
    - ☐ 10

19. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `5`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 4
    - ☐ 0
    - ☐ 16

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
2. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
3. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
   - ☐ des processus
   - ☐ les fichiers du disque
5. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
6. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 1 2 3 4 5 6 7
9. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = %d`

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`

14. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

15. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ créer un répertoire
- ☐ changer de répertoire courant

16. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 2
- ☐ 8 6 4 2 0

17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define N 3`
- ☐ `#define taille = 3`

18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

20. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 0
   - ☐ 8
   - ☐ 16
3. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!`
   - ☐ `!=`
   - ☐ $\neq$
4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 0 1 2`
5. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
6. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define taille = N`
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
8. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 16
   - ☐ 8
9. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
10. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6
11. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
13. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`
14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

    qu'est ce qui sera affiché ?
    - ☐ `** ** ** ** ** **`
    - ☐ `***** **** *** ***`
    - ☐ `**** **** **** ****`
    - ☐ `** *** **** *****`
15. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `loop i;`
    - ☐ `int k;`
16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
17. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
19. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 10
☐ 6
☐ 0
☐ 15

2. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ certaines données de la mémoire de travail
☐ les fichiers du disque
☐ en temps d'accès
☐ des processus

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

☐ la variable x vaut 5 et la variable y vaut 0
☐ le programme affiche "Faux"
☐ la variable y vaut 5
☐ la variable x vaut 0

4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `x = 5`
☐ le terminal affiche `5`
☐ le terminal affiche "Faux"
☐ le terminal affiche `x = 2`

5. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ un point-virgule manquant
☐ une accolade en trop
☐ une accolade manquante
☐ un point-virgule en trop

6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc -Wall prog.c -o prog.exe`
☐ `gcc -Wall prog.exe -o prog.c`
☐ `gcc prog.exe -Wall -o prog.c`
☐ `gcc prog.c -o -Wall prog.exe`

7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 3 0 1 2`
☐ `0 1 2 0 1 2`
☐ `0 0 1 1 2 2 3`
☐ `0 1 2 0 1 2 3`

8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 8 2
☐ 0 2 4 6 8
☐ 8 6 4 2 0
☐ 8 6 4 2

9. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1
☐ 1 2 3 4
☐ 4 3 2 1 0
☐ 0 1 2 3 4

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Majeur`
☐ `Majeur Mineur`
☐ `Mineur`
☐ rien

11. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `!`
    - ☐ `<>`
    - ☐ $\neq$
12. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ init
    - ☐ main
    - ☐ begin
13. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ voir des clips musicaux
14. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse lexicale
15. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` « `prinft` »
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'édition de liens
16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 4
    - ☐ 0
    - ☐ 8
17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche x
    - ☐ la variable x vaut 16
    - ☐ le programme affiche `****`
18. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
20. Le langage C est un langage
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
2. Le langage C est un langage
   - ☐ interprété
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé
3. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`
4. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[taille=5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `char tableau[5];`
5. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
6. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
11. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse lexicale
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 4`
    - ☐ `j = 5`
    - ☐ `j = %d`
    - ☐ `j = 0`

13. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

14. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

16. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :
- ☐ 42
- ☐ 0
- ☐ 1
- ☐ 6

17. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ main
    - ☐ init
    - ☐ begin

18. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

19. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ voir des clips musicaux
    - ☐ compiler un programme

20. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`

☐ `0 1 2 0 1 2 3`

☐ `0 1 2 3 0 1 2`

☐ `0 0 1 1 2 2 3`

2. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 8 6 4 2 0

☐ 0 2 4 6 8

☐ 8 6 4 2

☐ 8 2

3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

☐ `printf("x=%d et y=%d\n",x y);`

☐ `printf("x=%d et y=%d\n",x,y);`

☐ `printf("x=%d et y=%d\n,x,y");`

☐ `printf("x=%x et y=%y\n");`

4. Un registre du processeur est :

☐ une gamme de fréquence de fonctionnement du processeur

☐ une unité de calcul spécialisée de l'ordinateur

☐ un composant qui contient la liste des fichiers du système

☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`

☐ `#appart <stdlib.h>`

☐ `#include <studio.h>`

☐ `#include <stdio.h>`

6. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define taille = N`

☐ `#define taille = 3`

☐ `#define N =  3`

☐ `#define N 3`

7. Dans la commande gcc, l'option `-Wall` signifie :

☐ qu'il faut indenter le fichier source

☐ que l'on veut voir tous les avertissements

☐ qu'il faut lancer un déboggueur

☐ qu'on veut changer alétoirement de fond d'écran

8. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

☐ `new TP4`

☐ `kwrite TP4`

☐ `yppasswd`

☐ `mkdir TP4`

9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

☐ comportent une erreur qui sera détectée au cours de l'édition de lien

☐ ne comportent aucune erreur

☐ comportent une erreur qui ne sera pas détectée

☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

10. L'ordonnancement par tourniquet permet :

☐ d'afficher des ronds colorés à l'écran

☐ de doubler la mémoire disponible

☐ de ne pas perdre de temps avec la commutation de contexte

☐ d'entretenir l'illusion que les processus tournent en parallèle

11. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ en temps d'accès

☐ certaines données de la mémoire de travail

☐ les fichiers du disque

☐ des processus

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`

13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"

14. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre

15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc

16. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int[] new tableau(5);`
- ☐ `int toto[5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5

18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

20. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`

3. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`

4. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 16
   - ☐ 20
   - ☐ 6
   - ☐ 3

5. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ voir des clips musicaux

6. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien

9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `5`

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`

11. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé

12. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante

13. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1

14. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8
    - ☐ 0 2 4 6

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0

16. Le langage C est un langage
    - ☐ interprété
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé

17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 5`
    - ☐ `j = 0`
    - ☐ `j = 4`

19. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable x vaut 3

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
2. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `<>`
3. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
4. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
5. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `***** **** *** ***`
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
6. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
7. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Écrire des données sur le dique dur
8. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
9. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0

11. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
12. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc
13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `1 2 3 1 2`
14. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`
15. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 3
    - ☐ 6
    - ☐ 16
    - ☐ 20
16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 0
    - ☐ 15
    - ☐ 6
17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
20. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** −0,5 **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

2. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define N 3`

3. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois

4. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme
- ☐ voir des clips musicaux

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0

6. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur

7. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `new TP4`

8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

9. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 6
- ☐ 1
- ☐ 0

10. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 20
- ☐ 6
- ☐ 3
- ☐ 16

11. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ en temps d'accès
- ☐ les fichiers du disque

12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`

14. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée

15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`

16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8

18. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 4
- ☐ 16

19. Sous unix (ou linux), la commande `cd` permet de :

- ☐ ouvir un bureau partagé (common desktop)
- ☐ changer de répertoire courant
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ détruire un fichier

20. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int loop n;`
- ☐ `int %d;`
- ☐ `int k;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3

3. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire

4. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`

6. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique
   - ☐ l'analyse des entrées clavier

7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`

8. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!`

9. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire

10. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 6
   - ☐ 16
   - ☐ 3

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"

12. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 6
- ☐ 15
- ☐ 0

13. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`

16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose

17. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 8 2

19. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
20. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`

2. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop

3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`

4. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
- ☐ 20
- ☐ 16
- ☐ 3
- ☐ 6

5. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ détruire un fichier

6. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois

7. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle

8. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système

9. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
   - ☐ `int toto[taille=5];`

10. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

12. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

15. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
16. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`

17. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `<>`
    - ☐ $\neq$
    - ☐ `!`

18. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
    - ☐ compiler un programme
19. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
20. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `1 2 3 1 2`

☐ `0 1 2 0 1 2`

☐ `0 1 0 1 0 1`

☐ `0 0 1 1 2 2`

2. Le langage C est un langage

☐ composé

☐ compilé

☐ lu, écrit, parlé

☐ interprété

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0

☐ 4 3 2 1

☐ 0 1 2 3

☐ 0 1 2 3 4

4. Un registre du processeur est :

☐ un composant qui contient la liste des fichiers du système

☐ une gamme de fréquence de fonctionnement du processeur

☐ une unité de calcul spécialisée de l'ordinateur

☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

5. Quel est l'opérateur de différence en C :

☐ `!`

☐ `!=`

☐ `<>`

☐ $\neq$

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

☐ `Majeur Mineur`

☐ rien

☐ `Mineur`

☐ `Majeur`

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

☐ la variable x vaut 3

☐ la variable y vaut 5

☐ la variable x vaut 5 et la variable y vaut 3

☐ le programme affiche "Faux"

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`

☐ `#include <stdio.h>`

☐ `#appart <stdlib.h>`

☐ `#include <studio.h>`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ la variable x vaut $-\frac{1}{2}$

☐ le programme affiche x

☐ la variable x vaut 16

☐ le programme affiche `****`

10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 0

☐ 8

☐ 16

☐ 4

11. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
```

```
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`

12. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
15. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 15
- ☐ 10
- ☐ 0

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 0

17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

19. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
20. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `int %d;`
    - ☐ `loop i;`
    - ☐ `int k;`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0

2. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
   - ☐ `new TP4`

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1

4. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire

5. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

6. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
   - ☐ `1 2 1 2 3`

7. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de doubler la mémoire disponible

8. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`

9. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

10. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte

11. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`

12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 16
- ☐ 0

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`

14. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée
- ☐ un caractère interdit en C

15. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N =  3`
- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N 3`

17. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ en temps d'accès

18. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

20. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 8 2

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 4`

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `1 2 1 2 3`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1

4. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade manquante
- ☐ une accolade en trop
- ☐ un point-virgule manquant

5. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Écrire des données sur le dique dur
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Arriver à l'heure en cours

6. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ retourner un bloc

7. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher le contenu d'un fichier texte
- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire

8. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ $\neq$
- ☐ `!=`
- ☐ `<>`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ la variable x vaut 16

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

11. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0

12. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`
13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `5`
14. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
15. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
16. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 0
    - ☐ 6
    - ☐ 42
17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ `Majeur`
19. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire
   - ☐ créer un fichier texte

2. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`

3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 16
   - ☐ 8

4. L'ordonnancement par tourniquet permet :
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de doubler la mémoire disponible
   - ☐ de ne pas perdre de temps avec la commutation de contexte

5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1

6. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x y);`

7. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 16
   - ☐ 6
   - ☐ 20
   - ☐ 3

8. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé

10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` « `prinft` »
    - ☐ l'analyse sémantique
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier

11. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ include
    - ☐ begin
    - ☐ init

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`

13. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = 4`
- ☐ `j = %d`

14. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse sémantique
    - ☐ analyse syntaxique
    - ☐ analyse harmonique
15. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 0
17. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` ≪ `prinft` ≫ que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction
18. Le langage C est un langage
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété
19. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`

2. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
   - ☐ des processus

3. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

5. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`

6. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ composé
   - ☐ compilé

7. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`

8. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`

9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 2

10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ init
    - ☐ begin
    - ☐ main

11. Un registre du processeur est :
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
13. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte
14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
15. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
16. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
17. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
18. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int k;`
    - ☐ `int %d;`
19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 5`
    - ☐ `j = 0`
    - ☐ `j = %d`
    - ☐ `j = 4`
20. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`

2. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`

3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

4. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de ne pas perdre de temps avec la commutation de contexte

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x

6. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 4`

7. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur

8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7

9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier

10. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`

12. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

14. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

16. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule manquant
- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ une accolade manquante

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`

19. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int %d;`
- ☐ `int k;`
- ☐ `loop i;`
- ☐ `int loop n;`

20. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 10
☐ 0
☐ 15
☐ 6

2. Un programme en langage C doit comporter une et une seule définition de la fonction :

☐ include
☐ init
☐ main
☐ begin

3. Quel est l'opérateur de différence en C :

☐ `!`
☐ `<>`
☐ $\neq$
☐ `!=`

4. Le bus système sert à :

☐ Arriver à l'heure en cours
☐ Écrire des données sur le dique dur
☐ transporter les processus du tourniquet au processeur
☐ Transférer des données et intructions entre processeur et mémoire

5. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?

☐ une faute de frappe dans un appel de fonction
☐ une variable non déclarée
☐ une directive préprocesseur `#include` manquante
☐ un caractère interdit en C

6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 2 4 6 8
☐ 0 1 2 3 4 5 6
☐ 0 2 4 6
☐ 0 1 2 3 4 5 6 7

7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `5`
☐ le terminal affiche "Faux"
☐ le terminal affiche `x = 5`
☐ le terminal affiche `x = 2`

8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`
☐ `1 2 1 2 3`
☐ `0 0 0 1 1 1`
☐ `0 1 0 1 0 1 0 1`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ la variable x vaut $-\frac{1}{2}$
☐ le programme affiche `****`
☐ la variable x vaut 16
☐ le programme affiche x

10. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 8 6 4 2
☐ 0 2 4 6 8
☐ 8 6 4 2 0
☐ 8 2

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

12. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant

13. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3

15. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`

16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16

17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

18. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur

19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

20. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8

2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 0
- ☐ 15
- ☐ 6

3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `int %d;`
- ☐ `loop i;`

5. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int tab[] = 5;`
- ☐ `int toto[5];`
- ☐ `char tableau[5];`
- ☐ `int[] new tableau(5);`
- ☐ `int toto[taille=5];`

6. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse harmonique
- ☐ l'analyse des entrées clavier

7. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur

8. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse lexicale

9. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ main
- ☐ include
- ☐ begin

10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3

11. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`

12. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?

- ☐ une directive préprocesseur `#include` manquante
- ☐ une faute de frappe dans un appel de fonction
- ☐ une variable non déclarée
- ☐ un caractère interdit en C

13. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ en parallèle, chacun dans un registre
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble
- ☐ tour à tour, un petit peu à chaque fois

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `kwrite TP4`
- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `mkdir TP4`

15. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ le bus explose

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 0
- ☐ 16
- ☐ 8

17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

18. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ des processus

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"

20. Un fichier source est :

- ☐ un document de référence du système
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un fichier texte qui sera traduit en instructions processeur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante
2. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse des entrées clavier
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique
3. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
4. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique
5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
6. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
9. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
10. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0
12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
```

```
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

13. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
    - ☐ l'instruction qui met fin à un programme
14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
15. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 8 2

16. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
17. Un fichier source est :
    - ☐ un document qui doit être protégé
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

19. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
20. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ détruire un fichier
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`
2. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`
3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
4. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 16
   - ☐ 4
   - ☐ 0
5. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 3 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
6. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
7. Le langage C est un langage
   - ☐ interprété
   - ☐ composé
   - ☐ compilé
   - ☐ lu, écrit, parlé
8. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable x vaut 3
    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5
11. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
12. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche "Faux"

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`

15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%x et y=%y\n");`

16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0

17. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme
- ☐ voir des clips musicaux

18. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition

19. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ `!=`
- ☐ `<>`
- ☐ $\neq$

20. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 0
- ☐ 42
- ☐ 1

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche ****
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x

3. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse harmonique
   - ☐ analyse sémantique

4. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3

6. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 8
   - ☐ 4
   - ☐ 16

7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 0 1 2 3`

8. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`

9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
   - ☐ `new TP4`

10. Le langage C est un langage
    - ☐ compilé
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé

11. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours

12. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `int k;`
    - ☐ `loop i;`

13. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ en temps d'accès
    - ☐ les fichiers du disque
    - ☐ certaines données de la mémoire de travail

14. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse harmonique
- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique

15. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

16. Quel est l'opérateur de différence en C :

- ☐ `!=`
- ☐ `<>`
- ☐ `!`
- ☐ $\neq$

17. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`

18. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

19. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

2. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6

4. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 0
- ☐ 6
- ☐ 1

6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
7. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
8. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"

11. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ en parallèle, chacun dans un registre
    - ☐ chacun son tour, après que le processus précédent a terminé

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse lexicale

13. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée

14. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

15. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 8 6 4 2
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2 0

18. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`

19. Un registre du processeur est :
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ en temps d'accès
    - ☐ certaines données de la mémoire de travail
    - ☐ des processus
    - ☐ les fichiers du disque

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0

2. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`
   - ☐ `***** **** *** ***`

3. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `char tableau[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`

4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = 5`

5. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N 3`
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`

6. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ begin
   - ☐ init
   - ☐ include

7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`

8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`

9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8
   - ☐ 8 2
   - ☐ 8 6 4 2

10. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

11. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ `!`
- ☐ $\neq$
- ☐ `!=`

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`

14. L'ordonnancement par tourniquet permet :

- ☐ d'afficher des ronds colorés à l'écran
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ le programme affiche x

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `loop i;`
- ☐ `int loop n;`
- ☐ `int %d;`

17. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire

18. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 20
- ☐ 6
- ☐ 16
- ☐ 3

19. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse syntaxique
- ☐ analyse lexicale

20. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ détruire un fichier
- ☐ ouvir un bureau partagé (common desktop)
- ☐ changer de répertoire courant
- ☐ jouer de la musique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
   - ☐ `kwrite TP4`
2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'édition de liens
4. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
5. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1`
   - ☐ `1 2 3 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
8. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
9. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 4`
    - ☐ `j = 5`
    - ☐ `j = 0`
11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?

☐ `j = 0`

☐ `j = %d`

☐ `j = 5`

☐ `j = 4`

12. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ chacun son tour, après que le processus précédent a terminé

☐ tous ensemble

☐ tour à tour, un petit peu à chaque fois

☐ en parallèle, chacun dans un registre

13. L'ordonnancement par tourniquet permet :

☐ d'entretenir l'illusion que les processus tournent en parallèle

☐ de ne pas perdre de temps avec la commutation de contexte

☐ d'afficher des ronds colorés à l'écran

☐ de doubler la mémoire disponible

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 0

☐ 16

☐ 4

☐ 8

15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

☐ `printf("x=%x et y=%y\n");`

☐ `printf("x=%d et y=%d\n,x,y");`

☐ `printf("x=%d et y=%d\n",x y);`

☐ `printf("x=%d et y=%d\n",x,y);`

16. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`

☐ `#include <studio.h>`

☐ `#appart <stdlib.h>`

☐ `#include <stdio.h>`

17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3

☐ 4 3 2 1 0

☐ 4 3 2 1

☐ 0 1 2 3 4

18. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ les fichiers du disque

☐ en temps d'accès

☐ des processus

☐ certaines données de la mémoire de travail

19. Sous unix (ou linux), la commande `cd` permet de :

☐ ouvir un bureau partagé (common desktop)

☐ jouer de la musique

☐ changer de répertoire courant

☐ détruire un fichier

☐ récupérer un programme arrêté avec la commande `ab`

20. Un programme en langage C doit comporter une et une seule définition de la fonction :

☐ begin

☐ include

☐ init

☐ main

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
2. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
   - ☐ `new TP4`
3. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`
4. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`
5. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
6. L'ordonnancement par tourniquet permet :
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"
9. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
11. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `<>`
    - ☐ `!`
    - ☐ $\neq$
12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

15. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ begin
- ☐ init
- ☐ include

16. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux

17. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = %d`

19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque
- ☐ en temps d'accès
- ☐ des processus

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut lancer un déboggueur
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut indenter le fichier source
2. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ détruire un fichier
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`
3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien
5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = 0`
   - ☐ `j = %d`
   - ☐ `j = 4`
8. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`
10. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ une accolade en trop
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0

12. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
13. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 0
15. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ include
    - ☐ init
    - ☐ begin
16. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 3 0 1 2`
    - ☐ `0 1 2 0 1 2 3`
    - ☐ `0 1 2 0 1 2`
17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens
    - ☐ l'analyse des entrées clavier
18. Le langage C est un langage
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé
19. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ rien
20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 8
    - ☐ 4
    - ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5

2. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ créer un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire

3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

4. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 5`

6. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ begin
   - ☐ include
   - ☐ main

7. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire

8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée

9. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 16
   - ☐ 0
   - ☐ 8

10. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`

11. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 0
   - ☐ le terminal affiche 8
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse harmonique

13. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`

14. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 15
    - ☐ 10
    - ☐ 0
    - ☐ 6

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`

16. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6

17. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran

18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`

19. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur

20. Le langage C est un langage
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ composé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse sémantique
   - ☐ analyse harmonique
   - ☐ analyse syntaxique

2. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`

3. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ voir des clips musicaux

4. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
   - ☐ 8 2
   - ☐ 8 6 4 2

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0
   - ☐ la variable y vaut 5

6. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant

7. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `loop i;`
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `int k;`

8. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée

9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 3
    - ☐ la variable y vaut 5

11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6 8

12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
```

```
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ 0 1 2 0 1 2
- ☐ 0 1 2 0 1 2 3
- ☐ 0 0 1 1 2 2 3
- ☐ 0 1 2 3 0 1 2

13. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ 0 0 1 1 2 2
- ☐ 1 2 3 1 2
- ☐ 0 1 0 1 0 1
- ☐ 0 1 2 0 1 2

15. Quel est l'opérateur de différence en C :
    - ☐ <>
    - ☐ !
    - ☐ !=
    - ☐ ≠

16. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ **** **** **** ****
- ☐ ** *** **** *****
- ☐ ** ** ** ** ** **
- ☐ ***** **** *** ***

17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ 1 2 1 2 3
- ☐ 0 1 0 1 0 1 0 1
- ☐ 0 0 0 1 1 1
- ☐ 0 1 2 0 1 2

18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ en temps d'accès
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
2. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
3. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 0
   - ☐ 10
   - ☐ 6
   - ☐ 15
4. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
5. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `loop i;`
   - ☐ `int loop n;`
   - ☐ `int k;`
7. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 0`
   - ☐ `j = 4`
9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2`
10. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 1 2 3 4
11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
13. Le langage C est un langage
    - ☐ composé
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ compilé
14. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ récupérer un programme arrêté avec la commande `ab`
15. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 8
    - ☐ 16
    - ☐ 4
16. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ les fichiers du disque
17. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Mineur`
19. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 4
    - ☐ 0
    - ☐ 8
20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 0
    - ☐ le programme affiche "Faux"

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
2. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ main
   - ☐ include
   - ☐ begin
3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
5. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 8
   - ☐ 16
7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
9. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant

10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

11. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ en parallèle, chacun dans un registre
- ☐ tous ensemble
- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé

12. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ un point-virgule manquant
- ☐ une accolade manquante
- ☐ une accolade en trop

13. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 0
- ☐ 16
- ☐ 8

15. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`

16. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »`

- ☐ l'analyse sémantique
- ☐ l'analyse harmonique
- ☐ l'analyse des entrées clavier
- ☐ l'édition de liens

18. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16

19. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ transporter les processus du tourniquet au processeur
- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours

20. Sous unix (ou linux), la commande `cd` permet de :

- ☐ détruire un fichier
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ changer de répertoire courant
- ☐ ouvir un bureau partagé (common desktop)

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

2. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16

3. Un fichier source est :

- ☐ un document de référence du système
- ☐ un document qui doit être protégé
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document illisible pour les humains
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"

5. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 6
- ☐ 0
- ☐ 15

6. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements

7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

8. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ begin
- ☐ main
- ☐ include

9. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 4
- ☐ 16

10. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

11. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 3
- ☐ 16
- ☐ 20

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

13. Le langage C est un langage

- ☐ compilé
- ☐ composé
- ☐ lu, écrit, parlé
- ☐ interprété

14. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 42
- ☐ 0
- ☐ 1

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = N`
- ☐ `#define N 3`
- ☐ `#define N =  3`
- ☐ `#define taille = 3`

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ en temps d'accès
- ☐ les fichiers du disque
- ☐ des processus

20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
2. Un fichier source est :
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
   - ☐ un fichier texte qui sera traduit en instructions processeur
3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 8
   - ☐ 16
4. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
5. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int %d;`
   - ☐ `int k;`
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'édition de liens
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
10. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue
11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
12. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
13. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

☐ 0 1 2 3 4 5 6 7

☐ 0 2 4 6

☐ 0 1 2 3 4 5 6

☐ 0 2 4 6 8

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ 0 1 2 0 1 2 3

☐ 0 1 2 3 0 1 2

☐ 0 1 2 0 1 2

☐ 0 0 1 1 2 2 3

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ 1 2 3 1 2

☐ 0 1 2 0 1 2

☐ 0 0 1 1 2 2

☐ 0 1 0 1 0 1

16. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

☐ analyse lexicale

☐ analyse syntaxique

☐ analyse harmonique

☐ analyse sémantique

17. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

☐ `mkdir TP4`

☐ `yppasswd`

☐ `new TP4`

☐ `kwrite TP4`

18. Sur unix (ou linux), la commande `mkdir` permet de :

☐ changer de répertoire courant

☐ créer un fichier texte

☐ ouvrir un fichier texte

☐ créer un répertoire

19. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ chacun son tour, après que le processus précédent a terminé

☐ tous ensemble

☐ en parallèle, chacun dans un registre

☐ tour à tour, un petit peu à chaque fois

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ le programme affiche `****`

☐ le programme affiche x

☐ la variable x vaut 16

☐ la variable x vaut $-\frac{1}{2}$

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
2. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
5. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse harmonique
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
6. Le langage C est un langage
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ compilé
7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 2`
8. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
9. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
10. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5

☐ le programme affiche "Faux"

☐ la variable x vaut 5 et la variable y vaut 0

☐ la variable x vaut 0

12. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?

☐ une faute de frappe dans un appel de fonction

☐ une variable non déclarée

☐ un caractère interdit en C

☐ une directive préprocesseur `#include` manquante

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

☐ la case mémoire 8 contiendra 0

☐ la case mémoire 8 contiendra 16

☐ le bus explose

☐ le terminal affiche 8

14. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define taille = N`

☐ `#define taille = 3`

☐ `#define N 3`

☐ `#define N =  3`

15. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

☐ `int loop n;`

☐ `int k;`

☐ `loop i;`

☐ `int %d;`

16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0

☐ 4 3 2 1

☐ 0 1 2 3

☐ 0 1 2 3 4

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ 0 1 2 0 1 2

☐ 0 0 1 1 2 2

☐ 0 1 0 1 0 1

☐ 1 2 3 1 2

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

☐ `Majeur`

☐ `Mineur`

☐ `Majeur Mineur`

☐ rien

19. Quels calculs peut-on programmer en programmation structurée ?

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

☐ certains programmes sont de vrais plats de spaghetti

20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studio.h>`

☐ `#appart <stdlib.h>`

☐ `#include <studlib.h>`

☐ `#include <stdio.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`

3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`

4. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `kwrite TP4`
- ☐ `mkdir TP4`
- ☐ `new TP4`
- ☐ `yppasswd`

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `1 2 1 2 3`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`

6. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`
- ☐ `1 2 3 1 2`

7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`

8. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un fichier texte

9. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define N =  3`
- ☐ `#define taille = 3`
- ☐ `#define taille = N`

11. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade manquante
- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ un point-virgule manquant

12. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible

13. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ en temps d'accès
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque

14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?
- ☐ ***** **** *** ***
- ☐ **** **** **** ****
- ☐ ** *** **** *****
- ☐ ** ** ** ** ** **

15. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran

16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

17. Le langage C est un langage
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété

18. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

20. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`
2. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
3. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ begin
   - ☐ main
   - ☐ include
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 0 1 1 1`
5. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ la case mémoire 8 contiendra 0
6. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 8 2
   - ☐ 0 2 4 6 8
9. Un fichier source est :
   - ☐ un document illisible pour les humains
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
11. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
12. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3

13. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `char tableau[5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[taille=5];`
- ☐ `int toto[5];`
- ☐ `int[] new tableau(5);`

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `kwrite TP4`

15. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ un point-virgule manquant
- ☐ une accolade manquante

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"

17. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 3
- ☐ 20
- ☐ 16
- ☐ 6

18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`

19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur

20. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une variable non déclarée
- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante
- ☐ un caractère interdit en C

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
2. Le langage C est un langage
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ interprété
3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
4. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 8
   - ☐ 4
5. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
6. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
8. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
9. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?
    - ☐ `j = 5`
    - ☐ `j = 4`
    - ☐ `j = 0`
    - ☐ `j = %d`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"

12. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
13. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ l'instruction qui met fin à un programme
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire
14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`

15. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

17. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0

18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée

19. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ ouvrir un fichier texte
20. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`
- ☐ `**** **** **** ****`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0

2. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant

3. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique

4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?
- ☐ 0 0 1 1 2 2 3
- ☐ 0 1 2 3 0 1 2
- ☐ 0 1 2 0 1 2 3
- ☐ 0 1 2 0 1 2

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

6. L'ordonnancement par tourniquet permet :
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible

7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

8. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`

10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
    - ☐ afficher le contenu d'un fichier texte

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ le programme affiche ****
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16

12. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un chiffre binaire (0 ou 1)
13. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
14. Un registre du processeur est :
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
16. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ changer de répertoire courant
    - ☐ ouvrir un fichier texte
17. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 15
    - ☐ 0
    - ☐ 6
19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ les fichiers du disque
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ des processus
20. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur

2. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `1 2 3 1 2`

3. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 4
   - ☐ 8

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `int k;`
   - ☐ `loop i;`

5. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ récupérer un programme arrêté avec la commande `ab`

6. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`

7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`

8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 4`

9. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse sémantique
   - ☐ analyse syntaxique
   - ☐ analyse harmonique

10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ main
    - ☐ begin
    - ☐ include

11. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

12. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ les fichiers du disque
    - ☐ des processus
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
13. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien

14. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

15. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
16. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8
- ☐ 0 2 4 6

17. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
18. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une unité de calcul spécialisée de l'ordinateur
19. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

2. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8

3. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

   - ☐ certaines données de la mémoire de travail
   - ☐ les fichiers du disque
   - ☐ des processus
   - ☐ en temps d'accès

4. Sous unix (ou linux), la commande `cd` permet de :

   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)

5. Dans la commande gcc, l'option `-Wall` signifie :

   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :

   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Majeur`

7. Quel est l'opérateur de différence en C :

   - ☐ $\neq$
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`

9. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3

10. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

    - ☐ analyse lexicale
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse sémantique

11. Le bus système sert à :

    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur

12. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

    qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`

13. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`

15. Le langage C est un langage
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ composé
16. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ main
    - ☐ init
    - ☐ begin
17. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
18. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
    - ☐ `mkdir TP4`
19. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
20. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
2. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une variable non déclarée
   - ☐ un caractère interdit en C
   - ☐ une faute de frappe dans un appel de fonction
3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 6
   - ☐ 42
   - ☐ 0
6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
7. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
8. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
10. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0

12. L'ordonnancement par tourniquet permet :

- ☐ d'afficher des ronds colorés à l'écran
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ de doubler la mémoire disponible
- ☐ d'entretenir l'illusion que les processus tournent en parallèle

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
    produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 8
- ☐ 0

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`
- ☐ `#define N =  3`

16. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 0 1 1 2 2 3`

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`

18. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`

20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`

- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse harmonique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse harmonique

2. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

3. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 6
   - ☐ 42
   - ☐ 1
   - ☐ 0

4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ la case mémoire 8 contiendra 0

5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`

6. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!=`
   - ☐ `!`
   - ☐ $\neq$

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0

8. Le langage C est un langage
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ composé

9. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse harmonique

11. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
    - ☐ l'instruction qui met fin à un programme
12. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
13. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ détruire un fichier
14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 0
    - ☐ 4
    - ☐ 8
15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut 16
    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche x
    - ☐ le programme affiche `****`
17. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte
18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui ne sera pas détectée
19. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
20. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`

3. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

4. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

5. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 1
- ☐ 0
- ☐ 42
- ☐ 6

7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1

8. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ include
- ☐ main
- ☐ begin

9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

11. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte

12. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int %d;`

13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `x = 2`

14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0

16. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Arriver à l'heure en cours
- ☐ Écrire des données sur le dique dur
- ☐ transporter les processus du tourniquet au processeur

17. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un répertoire
- ☐ créer un fichier texte
- ☐ changer de répertoire courant
- ☐ ouvrir un fichier texte

18. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 3
- ☐ 16
- ☐ 6
- ☐ 20

19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`

20. Sous unix (ou linux), la commande `cd` permet de :

- ☐ détruire un fichier
- ☐ ouvir un bureau partagé (common desktop)
- ☐ jouer de la musique
- ☐ changer de répertoire courant
- ☐ récupérer un programme arrêté avec la commande `ab`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran
2. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 20
   - ☐ 3
   - ☐ 16
3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
4. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
5. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 16
   - ☐ 0
   - ☐ 8
7. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 2
10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche ****
- ☐ le programme affiche x

12. Un fichier source est :
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un fichier texte qui sera traduit en instructions processeur

13. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 8
- ☐ 0
- ☐ 16
- ☐ 4

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?
- ☐ 0 0 0 1 1 1
- ☐ 0 1 2 0 1 2
- ☐ 0 1 0 1 0 1 0 1
- ☐ 1 2 1 2 3

15. Un bit est :
- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)
- ☐ la longueur d'un mot mémoire
- ☐ l'instruction qui met fin à un programme

16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

17. Un registre du processeur est :
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur

18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 6
☐ 20
☐ 16
☐ 3

2. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4
☐ 4 3 2 1
☐ 1 2 3 4
☐ 4 3 2 1 0

3. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ tour à tour, un petit peu à chaque fois
☐ tous ensemble
☐ en parallèle, chacun dans un registre
☐ chacun son tour, après que le processus précédent a terminé

4. Sous unix (ou linux), la commande `ls` permet de :

☐ afficher la liste de fichiers contenus dans un répertoire
☐ afficher le contenu d'un fichier texte
☐ voir des clips musicaux
☐ compiler un programme

5. Sous unix (ou linux), la commande `cd` permet de :

☐ détruire un fichier
☐ récupérer un programme arrêté avec la commande `ab`
☐ ouvir un bureau partagé (common desktop)
☐ changer de répertoire courant
☐ jouer de la musique

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ des processus
☐ certaines données de la mémoire de travail
☐ les fichiers du disque
☐ en temps d'accès

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`
☐ `#include <studio.h>`
☐ `#include <stdio.h>`
☐ `#appart <stdlib.h>`

8. Un registre du processeur est :

☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
☐ un composant qui contient la liste des fichiers du système
☐ une unité de calcul spécialisée de l'ordinateur
☐ une gamme de fréquence de fonctionnement du processeur

9. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

☐ `printf("x=%d et y=%d\n,x,y");`
☐ `printf("x=%x et y=%y\n");`
☐ `printf("x=%d et y=%d\n",x,y);`
☐ `printf("x=%d et y=%d\n",x y);`

10. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 0
☐ 8
☐ 4
☐ 16

11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `0 0 1 1 2 2`
☐ `0 1 0 1 0 1`
☐ `1 2 3 1 2`
☐ `0 1 2 0 1 2`

12. Le bus système sert à :

- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ transporter les processus du tourniquet au processeur

13. Si cette erreur apparaît à la compilation :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?

- ☐ une faute de frappe dans un appel de fonction
- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée
- ☐ un caractère interdit en C

14. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose

15. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int[] new tableau(5);`
- ☐ `int tab[] = 5;`
- ☐ `char tableau[5];`
- ☐ `int toto[taille=5];`
- ☐ `int toto[5];`

16. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse sémantique

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0

18. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ le programme affiche x

20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'édition de liens
- ☐ l'analyse sémantique
- ☐ l'analyse harmonique
- ☐ l'analyse des entrées clavier

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
2. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 3 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1`
4. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
6. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
7. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
8. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
9. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur`
10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 3 0 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 2 0 1 2 3`
    - ☐ `0 0 1 1 2 2 3`
11. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
12. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `loop i;`
    - ☐ `int k;`
    - ☐ `int %d;`

13. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche x
- ☐ le programme affiche `****`

15. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ en temps d'accès
- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque

16. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

17. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ begin
- ☐ init
- ☐ include

18. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 0
- ☐ 8

19. Le langage C est un langage

- ☐ compilé
- ☐ composé
- ☐ lu, écrit, parlé
- ☐ interprété

20. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
2. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
3. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ rien
5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
6. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `!=`
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
8. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 4
    - ☐ 8
    - ☐ 16
11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
12. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n",x,y);`

13. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ la variable x vaut 16
    - ☐ le programme affiche x
    - ☐ le programme affiche `****`
    - ☐ la variable x vaut $-\frac{1}{2}$

15. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

16. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur

17. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫
    - ☐ l'analyse des entrées clavier
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens

18. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran

19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 4`
    - ☐ `j = 0`
    - ☐ `j = 5`

20. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ main
    - ☐ begin
    - ☐ init

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 points par réponse juste (unique) ; −0, 5 points par réponse fausse. Durée : 20 minutes.

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`

2. Les lignes
```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée

3. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ init
   - ☐ begin
   - ☐ include

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `int loop n;`
   - ☐ `loop i;`

5. Le code suivant :
```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```
   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur`

6. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur

7. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'analyse harmonique
   - ☐ l'édition de liens

8. Le code suivant :
```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```
   affichera :
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

9. Le langage C est un langage
   - ☐ composé
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ compilé

10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ voir des clips musicaux

11. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible

12. Le code suivant :
```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```
    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ rien

13. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade en trop
- ☐ une accolade manquante
- ☐ un point-virgule en trop
- ☐ un point-virgule manquant

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16
- ☐ le programme affiche x

15. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 0
- ☐ 8

16. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`

17. Un bit est :

- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme
- ☐ un battement d'horloge processeur

18. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une gamme de fréquence de fonctionnement du processeur

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`
   - ☐ `int loop n;`
2. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
3. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ include
   - ☐ main
   - ☐ init
4. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
5. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
7. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 3
   - ☐ 16
   - ☐ 20
   - ☐ 6
8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur`
9. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
10. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
11. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
12. Un registre du processeur est :
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
13. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue

- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

14. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade manquante

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 5`
    - ☐ `j = %d`
    - ☐ `j = 4`
    - ☐ `j = 0`

16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`

18. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse harmonique
    - ☐ analyse sémantique
    - ☐ analyse lexicale
    - ☐ analyse syntaxique

19. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ ne comportent aucune erreur

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
    - ☐ 16
    - ☐ 4
    - ☐ 0
    - ☐ 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int loop n;`
2. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
3. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 16
   - ☐ 6
   - ☐ 3
   - ☐ 20
4. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
5. Quel est l'opérateur de différence en C :
   - ☐ `!`
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!=`
6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
7. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `** ** ** ** ** **`
   - ☐ `**** **** **** ****`
   - ☐ `***** **** *** ***`
   - ☐ `** *** **** *****`
8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `new TP4`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
10. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 0 1 1 1`
    - ☐ `0 1 0 1 0 1 0 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `1 2 1 2 3`

12. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

13. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`

14. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ init
- ☐ include
- ☐ begin

15. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

16. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"

18. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ compiler un programme
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 3 0 1 2`

20. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
3. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
5. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien
8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 8
   - ☐ 0
   - ☐ 16
9. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse harmonique
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier
10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
    - ☐ afficher le contenu d'un fichier texte
11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ ne comportent aucune erreur

12. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 20
- ☐ 16
- ☐ 3

13. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse syntaxique

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

15. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur

16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5

18. Sous unix (ou linux), la commande `cd` permet de :

- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ jouer de la musique
- ☐ changer de répertoire courant
- ☐ détruire un fichier
- ☐ ouvir un bureau partagé (common desktop)

19. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade manquante
- ☐ une accolade en trop
- ☐ un point-virgule manquant
- ☐ un point-virgule en trop

20. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[taille=5];`
- ☐ `int[] new tableau(5);`
- ☐ `int toto[5];`
- ☐ `int tab[] = 5;`
- ☐ `char tableau[5];`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

2. Le langage C est un langage
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ interprété

3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ le programme affiche x

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int k;`
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int %d;`

9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16

10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
```

```
    produit = produit * serie[i];
  }
  printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 16
- ☐ 4

11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

12. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ init
- ☐ include
- ☐ begin

13. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ $\neq$
- ☐ `!`
- ☐ `!=`

14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`

15. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`
- ☐ `***** **** *** ***`

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

17. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2

19. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 1
- ☐ 6
- ☐ 0

20. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = %d`
3. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
4. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
5. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
6. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
8. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 0
   - ☐ 6
   - ☐ 15
   - ☐ 10
9. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule en trop
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3

11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`

12. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
13. Le langage C est un langage
    - ☐ interprété
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ composé
14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`

15. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
17. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
18. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`
19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
    - ☐ `new TP4`
20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ `Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 points par réponse juste (unique) ; −0,5 points par réponse fausse. Durée : 20 minutes.

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`

2. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 3
- ☐ 6
- ☐ 16
- ☐ 20

3. Sous unix (ou linux), la commande `cd` permet de :

- ☐ détruire un fichier
- ☐ changer de répertoire courant
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ ouvir un bureau partagé (common desktop)
- ☐ jouer de la musique

4. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.c -o -Wall prog.exe`

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

6. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ include
- ☐ init
- ☐ main

7. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre
- ☐ tous ensemble

8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`

9. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3

11. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

12. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = 4`

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`

17. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme

18. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ retourner un bloc
- ☐ sélectionner entre deux blocs à l'aide d'une condition

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse syntaxique
- ☐ analyse harmonique
- ☐ analyse sémantique
- ☐ analyse lexicale

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Écrire des données sur le dique dur
2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 8
   - ☐ 4
3. Le langage C est un langage
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ composé
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = %d`
5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 8 2
6. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
9. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 16
    - ☐ 4
    - ☐ 8

11. Sous unix (ou linux), la commande `cd` permet de :

- ☐ changer de répertoire courant
- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)
- ☐ détruire un fichier
- ☐ récupérer un programme arrêté avec la commande `ab`

12. Un fichier source est :

- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

13. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int %d;`
- ☐ `int loop n;`

15. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`

17. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

18. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ le programme affiche x
- ☐ la variable x vaut 16

19. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 3
- ☐ 6
- ☐ 16
- ☐ 20

20. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte
- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

   ☐ `#define N 3`

   ☐ `#define taille = 3`

   ☐ `#define N =  3`

   ☐ `#define taille = N`

2. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :

   ☐ 20

   ☐ 3

   ☐ 16

   ☐ 6

3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#include <stdio.h>`

   ☐ `#include <studio.h>`

   ☐ `#include <studlib.h>`

   ☐ `#appart <stdlib.h>`

4. Un bit est :

   ☐ un chiffre binaire (0 ou 1)

   ☐ un battement d'horloge processeur

   ☐ la longueur d'un mot mémoire

   ☐ l'instruction qui met fin à un programme

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?

   ☐ `0 0 0 1 1 1`

   ☐ `0 1 0 1 0 1 0 1`

   ☐ `1 2 1 2 3`

   ☐ `0 1 2 0 1 2`

6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

   ☐ `loop i;`

   ☐ `int %d;`

   ☐ `int k;`

   ☐ `int loop n;`

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#appart <stdlib.h>`

   ☐ `#include <stdio.h>`

   ☐ `#include <studlib.h>`

   ☐ `#include <studio.h>`

8. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   ☐ 4 3 2 1

   ☐ 4 3 2 1 0

   ☐ 1 2 3 4

   ☐ 0 1 2 3 4

9. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   ☐ 4 3 2 1

   ☐ 4 3 2 1 0

   ☐ 0 1 2 3

   ☐ 0 1 2 3 4

10. Un programme en langage C doit comporter une et une seule définition de la fonction :

    ☐ include

    ☐ begin

    ☐ main

    ☐ init

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    ☐ la variable x vaut 5 et la variable y vaut 0

    ☐ la variable y vaut 5

    ☐ la variable x vaut 0

    ☐ le programme affiche "Faux"

12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

13. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
14. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source
15. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
16. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4

17. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
18. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 2

20. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ le terminal affiche 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur

2. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 3
   - ☐ 16
   - ☐ 20

3. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`

4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = 5`
   - ☐ `j = %d`

5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ rien

6. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`
   - ☐ `#define N 3`

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`

8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

9. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`

10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `int %d;`
    - ☐ `int k;`
    - ☐ `loop i;`

11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 4
- ☐ 0

12. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
13. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ compiler un programme
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`

15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
16. L'ordonnancement par tourniquet permet :
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
17. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 8 2
- ☐ 0 2 4 6 8

19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 1 2 3 4

2. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

3. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ include
- ☐ init
- ☐ main

4. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

5. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ la division du programme en zones homogènes échoue

6. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ changer de répertoire courant
- ☐ créer un fichier texte
- ☐ ouvrir un fichier texte
- ☐ créer un répertoire

7. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'afficher des ronds colorés à l'écran

8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`

9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 8 2

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = N`
- ☐ `#define taille = 3`
- ☐ `#define N =  3`
- ☐ `#define N 3`

11. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 8
- ☐ 0

12. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une variable non déclarée
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une faute de frappe dans un appel de fonction

13. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 0
    - ☐ 8
    - ☐ 4

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?
    - ☐ `j = 4`
    - ☐ `j = 0`
    - ☐ `j = 5`
    - ☐ `j = %d`

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0

17. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti

18. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ `Mineur`

19. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale
    - ☐ analyse harmonique

20. Quel est l'opérateur de différence en C :
    - ☐ `<>`
    - ☐ `!=`
    - ☐ `!`
    - ☐ $\neq$

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 15
☐ 0
☐ 10
☐ 6

2. Un programme en langage C doit comporter une et une seule définition de la fonction :

☐ main
☐ include
☐ init
☐ begin

3. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

☐ `kwrite TP4`
☐ `mkdir TP4`
☐ `new TP4`
☐ `yppasswd`

4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studio.h>`
☐ `#include <stdio.h>`
☐ `#include <studlib.h>`
☐ `#appart <stdlib.h>`

5. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc -Wall prog.exe -o prog.c`
☐ `gcc prog.exe -Wall -o prog.c`
☐ `gcc -Wall prog.c -o prog.exe`
☐ `gcc prog.c -o -Wall prog.exe`

6. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

☐ `for(i=0;i<=5;i=i+1)`
☐ `for(i=0;i<5;i=i+1)`
☐ `for(i=1;i<=5;i=i+1)`
☐ `for(i=1;i<5;i=i+1)`

7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `x = 2`
☐ le terminal affiche `5`
☐ le terminal affiche "Faux"
☐ le terminal affiche `x = 5`

8. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade en trop
☐ un point-virgule manquant
☐ un point-virgule en trop
☐ une accolade manquante

9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 8 6 4 2
☐ 0 2 4 6 8
☐ 8 2
☐ 8 6 4 2 0

10. Sous unix (ou linux), la commande `cd` permet de :

☐ détruire un fichier
☐ changer de répertoire courant
☐ ouvir un bureau partagé (common desktop)
☐ jouer de la musique
☐ récupérer un programme arrêté avec la commande `ab`

11. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

☐ l'analyse sémantique
☐ l'édition de liens
☐ l'analyse harmonique
☐ l'analyse des entrées clavier

12. Quels calculs peut-on programmer en programmation structurée ?

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
☐ certains programmes sont de vrais plats de spaghetti
☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

13. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1

15. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6 7

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 8
    - ☐ 16
    - ☐ 4

17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `char tableau[5];`

18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée

19. Le bus système sert à :
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire

20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ rien

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
2. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `int toto[5];`
3. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
4. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
5. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 16
   - ☐ 0
   - ☐ 8
7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `1 2 3 1 2`
10. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 16
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le bus explose
    - ☐ le terminal affiche 8

11. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur

12. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée

13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 1 2 3 4

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 3
    - ☐ le programme affiche "Faux"

15. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé

16. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ compilé
    - ☐ composé

17. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant

18. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop

19. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ rien
    - ☐ `Majeur Mineur`

20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut lancer un déboggueur
2. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
3. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
4. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
5. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0
6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 3
   - ☐ 20
   - ☐ 16
   - ☐ 6
7. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur
8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 4
   - ☐ 8
   - ☐ 0
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 3
10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ main
    - ☐ init
    - ☐ include
11. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`

13. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte

14. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`

15. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 2

16. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 42
- ☐ 6
- ☐ 1

17. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `int %d;`

18. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0

19. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme

20. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

☐ `1 2 1 2 3`
☐ `0 1 0 1 0 1 0 1`
☐ `0 1 2 0 1 2`
☐ `0 0 0 1 1 1`

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <stdio.h>`
☐ `#appart <stdlib.h>`
☐ `#include <studio.h>`
☐ `#include <studlib.h>`

3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2`
☐ `0 1 0 1 0 1`
☐ `0 0 1 1 2 2`
☐ `1 2 3 1 2`

4. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

☐ `printf("x=%x et y=%y\n");`
☐ `printf("x=%d et y=%d\n,x,y");`
☐ `printf("x=%d et y=%d\n",x,y);`
☐ `printf("x=%d et y=%d\n",x y);`

5. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 16
☐ 4
☐ 8
☐ 0

6. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ un point-virgule en trop
☐ une accolade manquante
☐ une accolade en trop
☐ un point-virgule manquant

7. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 6
☐ 10
☐ 0
☐ 15

8. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc prog.c -o -Wall prog.exe`
☐ `gcc prog.exe -Wall -o prog.c`
☐ `gcc -Wall prog.exe -o prog.c`
☐ `gcc -Wall prog.c -o prog.exe`

9. Le bus système sert à :

☐ transporter les processus du tourniquet au processeur
☐ Transférer des données et intructions entre processeur et mémoire
☐ Écrire des données sur le dique dur
☐ Arriver à l'heure en cours

10. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3
☐ 4 3 2 1
☐ 0 1 2 3 4
☐ 4 3 2 1 0

11. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ en temps d'accès
☐ des processus
☐ certaines données de la mémoire de travail
☐ les fichiers du disque

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse syntaxique

13. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre

14. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`

16. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ main
    - ☐ include
    - ☐ begin

17. Le langage C est un langage
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ composé

18. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`

19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le terminal affiche 8
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 16

20. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ récupérer un programme arrêté avec la commande `ab`
2. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2
   - ☐ 8 2
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define taille = N`
4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
5. Le langage C est un langage
   - ☐ interprété
   - ☐ compilé
   - ☐ composé
   - ☐ lu, écrit, parlé
6. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `char tableau[5];`
7. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
8. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
9. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de ne pas perdre de temps avec la commutation de contexte
10. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
11. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`
12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 0
    - ☐ la case mémoire 8 contiendra 16
    - ☐ le bus explose
    - ☐ le terminal affiche 8
13. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ ouvrir un fichier texte

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `1 2 1 2 3`

15. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`

17. Quel est l'opérateur de différence en C :

- ☐ `!=`
- ☐ $\neq$
- ☐ `!`
- ☐ `<>`

18. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 42
- ☐ 6
- ☐ 1

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`

20. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 6
- ☐ 0
- ☐ 15

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ interprété
   - ☐ compilé
2. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3
4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
5. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 0
   - ☐ 8
   - ☐ 16
6. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
7. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
8. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = %d`
10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`

12. Un fichier source est :
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système
    - ☐ un document illisible pour les humains
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`

14. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ `Majeur Mineur`

16. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`

17. Quel est l'opérateur de différence en C :
    - ☐ `!`
    - ☐ `!=`
    - ☐ $\neq$
    - ☐ `<>`

18. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`

19. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur

20. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :
- ☐ 16
- ☐ 3
- ☐ 20
- ☐ 6

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`

3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4

4. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`

5. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8

6. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 10
   - ☐ 0
   - ☐ 15

7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ voir des clips musicaux
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme

8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 0 1 1 1`

9. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

10. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop

11. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse syntaxique
    - ☐ analyse lexicale

12. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 0
- ☐ 16

15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x,y);`

16. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

20. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ un point-virgule manquant
2. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
4. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
5. Le langage C est un langage
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ interprété
6. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `loop i;`
   - ☐ `int loop n;`
   - ☐ `int k;`
9. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
10. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
11. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
    - ☐ `kwrite TP4`
    - ☐ `new TP4`
12. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4

14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N 3`
- ☐ `#define N =  3`

16. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[taille=5];`
- ☐ `int tab[] = 5;`
- ☐ `int[] new tableau(5);`
- ☐ `int toto[5];`
- ☐ `char tableau[5];`

18. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti

19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

20. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 16
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 3

5. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 0
- ☐ 8

6. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`

8. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue

9. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `int k;`
    - ☐ `loop i;`
    - ☐ `int loop n;`
11. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ rien
12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 0
    - ☐ 4
    - ☐ 16
13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 1 1 2 2`
    - ☐ `1 2 3 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1`
14. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
15. Le langage C est un langage
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
    - ☐ composé
16. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
    - ☐ créer un répertoire
17. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse lexicale
    - ☐ analyse syntaxique
    - ☐ analyse harmonique
18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
19. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante
20. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ rien
☐ `Mineur`
☐ `Majeur`
☐ `Majeur Mineur`

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`
☐ `#include <studio.h>`
☐ `#include <stdio.h>`
☐ `#appart <stdlib.h>`

3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1
☐ 4 3 2 1 0
☐ 0 1 2 3 4
☐ 1 2 3 4

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ le programme affiche `****`
☐ le programme affiche x
☐ la variable x vaut 16
☐ la variable x vaut $-\frac{1}{2}$

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

☐ `j = 5`
☐ `j = %d`
☐ `j = 4`
☐ `j = 0`

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

☐ la variable y vaut 5
☐ la variable x vaut 0
☐ la variable x vaut 5 et la variable y vaut 0
☐ le programme affiche "Faux"

7. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

☐ certaines données de la mémoire de travail
☐ des processus
☐ en temps d'accès
☐ les fichiers du disque

8. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

☐ `***** **** *** ***`
☐ `** ** ** ** ** **`
☐ `** *** **** *****`
☐ `**** **** **** ****`

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

10. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
11. Quel est l'opérateur de différence en C :
    - ☐ $\neq$
    - ☐ `!=`
    - ☐ `!`
    - ☐ `<>`
12. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.c -o -Wall prog.exe`
13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

14. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
15. Un bit est :
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ 0 0 1 1 2 2
- ☐ 1 2 3 1 2
- ☐ 0 1 0 1 0 1
- ☐ 0 1 2 0 1 2

18. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ retourner un bloc
19. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le terminal affiche 8
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 0
    - ☐ la case mémoire 8 contiendra 16
20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur

2. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 16

3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int loop n;`
   - ☐ `int k;`
   - ☐ `loop i;`

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 5`
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = %d`

6. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse sémantique

7. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès

8. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système

9. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`

11. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 0 1 1 1`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`

13. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ interprété
    - ☐ compilé
    - ☐ composé

14. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une directive préprocesseur `#include` manquante

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`

16. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ jouer de la musique
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant

17. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 2

19. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ le programme affiche "Faux"

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur

2. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :

   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1

3. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`

4. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`

5. Sur unix (ou linux), la commande `mkdir` permet de :

   - ☐ créer un fichier texte
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant

6. Le bus système sert à :

   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur

7. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :

   - ☐ 42
   - ☐ 0
   - ☐ 1
   - ☐ 6

8. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 5`

9. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

   - ☐ les fichiers du disque
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès

10. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :

    - ☐ 8 6 4 2 0
    - ☐ 0 2 4 6 8
    - ☐ 8 6 4 2
    - ☐ 8 2

11. Sous unix (ou linux), la commande `ls` permet de :

    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire

12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ 0 1 2 0 1 2 3
- ☐ 0 0 1 1 2 2 3
- ☐ 0 1 2 3 0 1 2
- ☐ 0 1 2 0 1 2

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16

14. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ Majeur
- ☐ Majeur Mineur
- ☐ Mineur

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ Majeur Mineur
- ☐ Mineur
- ☐ rien
- ☐ Majeur

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 8
- ☐ 0

17. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

18. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée

19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3

20. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
2. Un fichier source est :
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
3. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = 0`
   - ☐ `j = 4`
   - ☐ `j = %d`
5. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 0 1 2`
6. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
7. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[5];`
   - ☐ `char tableau[5];`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`
   - ☐ `0 0 0 1 1 1`
10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`

☐ la variable x vaut 16

☐ la variable x vaut $-\frac{1}{2}$

☐ le programme affiche x

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

☐ `#include <studlib.h>`

☐ `#include <studio.h>`

☐ `#appart <stdlib.h>`

☐ `#include <stdio.h>`

12. Sur unix (ou linux), la commande `mkdir` permet de :

☐ ouvrir un fichier texte

☐ changer de répertoire courant

☐ créer un répertoire

☐ créer un fichier texte

13. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Majeur`

☐ rien

☐ `Mineur`

☐ `Majeur Mineur`

14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc prog.c -o -Wall prog.exe`

☐ `gcc -Wall prog.c -o prog.exe`

☐ `gcc prog.exe -Wall -o prog.c`

☐ `gcc -Wall prog.exe -o prog.c`

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Majeur Mineur`

☐ `Mineur`

☐ `Majeur`

☐ rien

16. Le bus système sert à :

☐ Transférer des données et intructions entre processeur et mémoire

☐ Arriver à l'heure en cours

☐ Écrire des données sur le dique dur

☐ transporter les processus du tourniquet au processeur

17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

☐ `#define taille = 3`

☐ `#define N 3`

☐ `#define N =  3`

☐ `#define taille = N`

18. Quels calculs peut-on programmer en programmation structurée ?

☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

☐ certains programmes sont de vrais plats de spaghetti

19. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 20

☐ 6

☐ 3

☐ 16

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3

☐ 0 1 2 3 4

☐ 4 3 2 1 0

☐ 4 3 2 1

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ le bus explose
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0

2. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`

3. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ main
   - ☐ include
   - ☐ init

4. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4

5. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ retourner un bloc

6. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire

7. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 16
   - ☐ 3
   - ☐ 20

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x
   - ☐ le programme affiche `****`

9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`

10. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 2`

12. L'ordonnancement par tourniquet permet :
    - ☐ de doubler la mémoire disponible
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int k;`

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 16
- ☐ 8
- ☐ 4
- ☐ 0

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 0 2 4 6 8

19. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres
2. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
3. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
4. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
5. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
6. Un fichier source est :
   - ☐ un document qui doit être protégé
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document illisible pour les humains
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document de référence du système
7. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
8. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
9. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!=`
   - ☐ $\neq$
   - ☐ `!`
10. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé
    - ☐ interprété
11. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 8 6 4 2
    - ☐ 0 2 4 6 8
    - ☐ 8 2
    - ☐ 8 6 4 2 0
12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `0 0 1 1 2 2`
    - ☐ `1 2 3 1 2`

13. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `int %d;`

14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

17. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

18. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `new TP4`
- ☐ `kwrite TP4`

19. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`
- ☐ `#define N =  3`

20. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante

2. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`
   - ☐ l'édition de liens
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique

3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6
   - ☐ 0 2 4 6 8

4. Quel est l'opérateur de différence en C :
   - ☐ ≠
   - ☐ `!=`
   - ☐ `<>`
   - ☐ `!`

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`

6. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`

8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`

9. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `1 2 1 2 3`
    - ☐ `0 0 0 1 1 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 0 1 0 1 0 1`

11. Le langage C est un langage
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 0
    - ☐ 8
    - ☐ 4

13. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant

14. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 3
    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable y vaut 5

16. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 5`
    - ☐ `j = 0`
    - ☐ `j = 4`

17. Un registre du processeur est :
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

18. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ changer de répertoire courant

19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0

20. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

    qu'est ce qui sera affiché ?
    - ☐ `**** **** **** ****`
    - ☐ `** ** ** ** ** **`
    - ☐ `** *** **** *****`
    - ☐ `***** **** *** ***`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique

2. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 6
   - ☐ 1
   - ☐ 0
   - ☐ 42

3. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8

4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 0`
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 5`

5. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5

7. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!`
   - ☐ $\neq$
   - ☐ `!=`

8. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur

9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`

11. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `kwrite TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`

12. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse des entrées clavier
- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse harmonique

13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`

14. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche x

16. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

17. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ des processus
- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque

18. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

19. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée

20. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ changer de répertoire courant
- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ créer un répertoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 0 0 1 1 1`
3. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
4. Un registre du processeur est :
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`
7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
8. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `new TP4`
9. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ composé
   - ☐ interprété
   - ☐ compilé
10. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ include
    - ☐ main
    - ☐ begin
11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`
12. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.exe -Wall -o prog.c`

13. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = 0`
    - ☐ `j = 4`
    - ☐ `j = %d`
    - ☐ `j = 5`

15. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 10
    - ☐ 15
    - ☐ 0
    - ☐ 6

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"

17. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

18. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3

19. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

20. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 16
    - ☐ 4
    - ☐ 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

2. Quel est l'opérateur de différence en C :

- ☐ `!=`
- ☐ `<>`
- ☐ $\neq$
- ☐ `!`

3. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 20
- ☐ 16
- ☐ 3

4. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

5. Un fichier source est :

- ☐ un document de référence du système
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains

6. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte

7. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ en temps d'accès

8. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 8
- ☐ 4

9. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une faute de frappe dans un appel de fonction
- ☐ un caractère interdit en C
- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée

10. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`
- ☐ `***** **** *** ***`

11. Après exécution du programme :

```
1   lecture 8 r0
2   valeur 3 r1
3   mult r1 r0
4   valeur 1 r2
5   add r2 r0
6   ecriture r0 8
7   stop
8   5
```

- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16

12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int[] new tableau(5);`
- ☐ `int toto[5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`

15. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int %d;`
- ☐ `int k;`
- ☐ `int loop n;`
- ☐ `loop i;`

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10   ...
11   int main() {
12       int x = 5;
13
14       x = 3 * x + 1;
15
16       ...
17   }
```

- ☐ le programme affiche x
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$

18. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

19. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`

20. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `kwrite TP4`
- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `new TP4`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 4
- ☐ 0
- ☐ 8

2. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

3. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse syntaxique
- ☐ analyse sémantique
- ☐ analyse harmonique
- ☐ analyse lexicale

4. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme

5. Un fichier source est :

- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document de référence du système
- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`

7. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte

8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int k;`
- ☐ `int %d;`
- ☐ `loop i;`
- ☐ `int loop n;`

9. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ retourner un bloc
- ☐ mettre les blocs en séquence les uns à la suite des autres

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N =  3`
- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`

11. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ include
- ☐ main
- ☐ begin

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 4`

13. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

14. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur

15. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 10
- ☐ 0
- ☐ 6

16. Le bus système sert à :

- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur
- ☐ Écrire des données sur le dique dur
- ☐ Transférer des données et intructions entre processeur et mémoire

17. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

18. Le langage C est un langage

- ☐ interprété
- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ composé

19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ transporter les processus du tourniquet au processeur

2. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = %d`

5. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier

6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme

7. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int tab[] = 5;`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int[] new tableau(5);`

8. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse sémantique

9. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ certaines données de la mémoire de travail

10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ le programme affiche x
- ☐ la variable x vaut 16

12. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
14. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
15. Le langage C est un langage
    - ☐ interprété
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé
16. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 3 0 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 0 1 2 3`
18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`
19. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une variable non déclarée
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ un caractère interdit en C
20. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x

2. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 6
   - ☐ 3
   - ☐ 16
   - ☐ 20

3. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 1
   - ☐ 42
   - ☐ 0
   - ☐ 6

4. Si cette erreur apparaît à la compilation :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante

5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8

6. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ begin
   - ☐ main
   - ☐ include

7. Le bus système sert à :
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire

8. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

9. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%x et y=%y\n");`

10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5

12. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`

13. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 4`

16. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire

17. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un fichier texte
    - ☐ changer de répertoire courant
    - ☐ ouvrir un fichier texte
    - ☐ créer un répertoire

18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`

20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse lexicale
    - ☐ analyse harmonique
    - ☐ analyse sémantique
    - ☐ analyse syntaxique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

3. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade manquante
- ☐ une accolade en trop
- ☐ un point-virgule manquant

4. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0

5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`

6. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`

7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

8. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ d'afficher des ronds colorés à l'écran
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ de ne pas perdre de temps avec la commutation de contexte

9. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

10. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`

12. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document de référence du système
    - ☐ un document qui doit être protégé

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose
- ☐ le terminal affiche 8

14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5

16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`

17. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
    - ☐ une directive préprocesseur `#include` manquante

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`

19. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`

20. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
2. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `***** **** *** ***`
   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`
   - ☐ `**** **** **** ****`
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
4. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`
5. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int k;`
   - ☐ `int %d;`
   - ☐ `loop i;`
   - ☐ `int loop n;`
6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
7. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
8. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ certaines données de la mémoire de travail
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ des processus
9. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ retourner un bloc
10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
11. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 0
    - ☐ 6
    - ☐ 42
12. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 20
    - ☐ 3
    - ☐ 16
    - ☐ 6

13. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 0 1 1 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `1 2 1 2 3`
    - ☐ `0 1 0 1 0 1 0 1`

15. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur`

16. Le langage C est un langage
    - ☐ interprété
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ composé

17. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ init
    - ☐ include
    - ☐ begin

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`

19. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop
    - ☐ une accolade en trop
    - ☐ une accolade manquante

20. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche `x = 5`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :

   ☐ 3

   ☐ 16

   ☐ 6

   ☐ 20

2. Sur unix (ou linux), la commande `mkdir` permet de :

   ☐ créer un répertoire

   ☐ créer un fichier texte

   ☐ changer de répertoire courant

   ☐ ouvrir un fichier texte

3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

   ☐ sélectionner entre deux blocs à l'aide d'une condition

   ☐ répéter un bloc tant qu'une condition est vérifée

   ☐ retourner un bloc

   ☐ mettre les blocs en séquence les uns à la suite des autres

4. Dans la commande gcc, l'option `-Wall` signifie :

   ☐ qu'on veut changer alétoirement de fond d'écran

   ☐ qu'il faut lancer un déboggueur

   ☐ que l'on veut voir tous les avertissements

   ☐ qu'il faut indenter le fichier source

5. Le langage C est un langage

   ☐ composé

   ☐ compilé

   ☐ lu, écrit, parlé

   ☐ interprété

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#include <studio.h>`

   ☐ `#include <studlib.h>`

   ☐ `#appart <stdlib.h>`

   ☐ `#include <stdio.h>`

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :

   ☐ `Majeur Mineur`

   ☐ `Mineur`

   ☐ `Majeur`

   ☐ rien

8. Le bus système sert à :

   ☐ transporter les processus du tourniquet au processeur

   ☐ Arriver à l'heure en cours

   ☐ Transférer des données et intructions entre processeur et mémoire

   ☐ Écrire des données sur le dique dur

9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   ☐ la case mémoire 8 contiendra 0

   ☐ le bus explose

   ☐ la case mémoire 8 contiendra 16

   ☐ le terminal affiche 8

10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :

    ☐ rien

    ☐ `Majeur`

    ☐ `Mineur`

    ☐ `Majeur Mineur`

11. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

    ☐ `int k;`

    ☐ `loop i;`

    ☐ `int %d;`

    ☐ `int loop n;`

12. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

    ☐ `int toto[taille=5];`

    ☐ `int toto[5];`

    ☐ `int[] new tableau(5);`

    ☐ `char tableau[5];`

    ☐ `int tab[] = 5;`

13. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

14. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 2

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 1
- ☐ 42
- ☐ 0

16. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ un point-virgule manquant
- ☐ une accolade manquante

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 16
- ☐ 0

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`

19. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ `!=`
- ☐ ≠
- ☐ `<>`

20. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
2. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
3. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
4. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 10
   - ☐ 0
   - ☐ 15
   - ☐ 6
5. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document illisible pour les humains
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
9. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse harmonique
   - ☐ analyse lexicale
10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`

11. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 4
- ☐ 0

12. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
13. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ main
    - ☐ init
    - ☐ begin
    - ☐ include
14. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur
15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
17. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[taille=5];`
18. Un registre du processeur est :
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une gamme de fréquence de fonctionnement du processeur
19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`

20. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut indenter le fichier source
    - ☐ qu'il faut lancer un déboggueur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé

2. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6

3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ 0 0 1 1 2 2 3
   - ☐ 0 1 2 3 0 1 2
   - ☐ 0 1 2 0 1 2
   - ☐ 0 1 2 0 1 2 3

4. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

5. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ composé

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = 4`

8. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme

9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8

10. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0

11. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de doubler la mémoire disponible

12. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `kwrite TP4`

13. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%x et y=%y\n");`

14. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 0
- ☐ 16

15. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Majeur`

16. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`

17. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur

18. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`

19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2
- ☐ 8 2

20. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 3
- ☐ 20
- ☐ 6
- ☐ 16

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 10
   - ☐ 0
   - ☐ 15
   - ☐ 6
3. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
4. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ $\neq$
   - ☐ `!`
   - ☐ `!=`
5. Le langage C est un langage
   - ☐ interprété
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé
6. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
   - ☐ un point-virgule en trop
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
8. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 2 0 1 2`
10. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = N`
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
12. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ de ne pas perdre de temps avec la commutation de contexte

13. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ en temps d'accès

14. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

15. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int %d;`
- ☐ `int k;`

16. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ main
- ☐ init
- ☐ include

17. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`

18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

20. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ compiler un programme

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
2. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
4. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
5. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ init
   - ☐ main
   - ☐ include
6. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `***** **** *** ***`
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`
   - ☐ `** ** ** ** ** **`
7. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
9. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de doubler la mémoire disponible
10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
11. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
12. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

13. Le langage C est un langage
    - ☐ lu, écrit, parlé
    - ☐ composé
    - ☐ compilé
    - ☐ interprété
14. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
15. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `loop i;`
16. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 1
- ☐ 42
- ☐ 6
- ☐ 0

17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

18. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `!`
    - ☐ $\neq$
    - ☐ `<>`
19. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 16
- ☐ 8
- ☐ 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
2. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
3. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
4. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `1 2 1 2 3`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0

7. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse harmonique
   - ☐ analyse lexicale
   - ☐ analyse sémantique
8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16

9. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

10. Le langage C est un langage
    - ☐ composé
    - ☐ interprété
    - ☐ compilé
    - ☐ lu, écrit, parlé
11. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 6
- ☐ 10
- ☐ 15

12. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

13. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 4`

14. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule manquant
- ☐ une accolade manquante
- ☐ un point-virgule en trop
- ☐ une accolade en trop

15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`

16. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ init
- ☐ include
- ☐ begin

17. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme

18. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 4
- ☐ 8

20. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ en temps d'accès
- ☐ des processus
- ☐ les fichiers du disque

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 4
- ☐ 16
- ☐ 8

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

3. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte

4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition

5. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8
- ☐ 0 2 4 6

6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 1
- ☐ 0
- ☐ 6

7. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`

8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 2

9. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

10. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1

☐ 4 3 2 1 0

☐ 1 2 3 4

☐ 0 1 2 3 4

11. Un bit est :

☐ l'instruction qui met fin à un programme

☐ la longueur d'un mot mémoire

☐ un battement d'horloge processeur

☐ un chiffre binaire (0 ou 1)

12. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?

☐ une variable non déclarée

☐ une directive préprocesseur `#include` manquante

☐ une faute de frappe dans un appel de fonction

☐ un caractère interdit en C

13. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

☐ `int k;`

☐ `int loop n;`

☐ `loop i;`

☐ `int %d;`

14. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

☐ 0

☐ 10

☐ 6

☐ 15

15. Sur unix (ou linux), la commande `mkdir` permet de :

☐ changer de répertoire courant

☐ créer un répertoire

☐ ouvrir un fichier texte

☐ créer un fichier texte

16. Quels calculs peut-on programmer en programmation structurée ?

☐ certains programmes sont de vrais plats de spaghetti

☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 1 2 3 4

☐ 1 2 3 4

☐ 4 3 2 1 0

☐ 4 3 2 1

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

☐ `int toto[taille=5];`

☐ `int tab[] = 5;`

☐ `char tableau[5];`

☐ `int[] new tableau(5);`

☐ `int toto[5];`

19. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 16

☐ 4

☐ 0

☐ 8

20. Un fichier source est :

☐ un document illisible pour les humains

☐ un document de référence du système

☐ un fichier texte qui sera traduit en instructions processeur

☐ un document qui doit être protégé

☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

2. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 0
   - ☐ 6
   - ☐ 1
   - ☐ 42

3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 0`
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 4`

4. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!`
   - ☐ $\neq$
   - ☐ `!=`

5. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`
   - ☐ `#define N 3`

6. Le bus système sert à :
   - ☐ Arriver à l'heure en cours
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Écrire des données sur le dique dur

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0

8. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `yppasswd`
   - ☐ `new TP4`
   - ☐ `kwrite TP4`
   - ☐ `mkdir TP4`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x

10. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3

11. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 15
- ☐ 0
- ☐ 6

12. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
            ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = %d`

15. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`

16. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

17. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un répertoire

18. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ include
- ☐ main
- ☐ begin
- ☐ init

19. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

20. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

   ☐ `int %d;`

   ☐ `loop i;`

   ☐ `int k;`

   ☐ `int loop n;`

2. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   ☐ rien

   ☐ `Mineur`

   ☐ `Majeur Mineur`

   ☐ `Majeur`

3. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

   ☐ `for(i=1;i<5;i=i+1)`

   ☐ `for(i=0;i<5;i=i+1)`

   ☐ `for(i=0;i<=5;i=i+1)`

   ☐ `for(i=1;i<=5;i=i+1)`

4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   ☐ `Mineur`

   ☐ rien

   ☐ `Majeur`

   ☐ `Majeur Mineur`

5. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   ☐ le terminal affiche 8

   ☐ le bus explose

   ☐ la case mémoire 8 contiendra 0

   ☐ la case mémoire 8 contiendra 16

6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :

   ☐ 42

   ☐ 1

   ☐ 0

   ☐ 6

7. L'ordonnancement par tourniquet permet :

   ☐ d'entretenir l'illusion que les processus tournent en parallèle

   ☐ de ne pas perdre de temps avec la commutation de contexte

   ☐ d'afficher des ronds colorés à l'écran

   ☐ de doubler la mémoire disponible

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   ☐ la variable y vaut 5

   ☐ la variable x vaut 3

   ☐ le programme affiche "Faux"

   ☐ la variable x vaut 5 et la variable y vaut 3

9. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

   ☐ `kwrite TP4`

   ☐ `mkdir TP4`

   ☐ `yppasswd`

   ☐ `new TP4`

10. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

    ☐ `#include <studio.h>`

    ☐ `#include <stdio.h>`

    ☐ `#appart <stdlib.h>`

    ☐ `#include <studlib.h>`

11. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 16
- ☐ 6
- ☐ 20
- ☐ 3

12. Sous unix (ou linux), la commande `cd` permet de :

- ☐ changer de répertoire courant
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ détruire un fichier
- ☐ jouer de la musique
- ☐ ouvir un bureau partagé (common desktop)

13. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

14. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

16. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique

17. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`

18. Un fichier source est :

- ☐ un document illisible pour les humains
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document qui doit être protégé

19. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme

20. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ changer de répertoire courant

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
2. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
3. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
4. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 8 2
5. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
6. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
8. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ init
   - ☐ main
   - ☐ include
9. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
   - ☐ voir des clips musicaux
10. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule manquant
    - ☐ une accolade manquante
    - ☐ une accolade en trop
    - ☐ un point-virgule en trop
11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 2 3 0 1 2`
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 0 1 2 3`
12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ rien

13. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`
- ☐ `new TP4`

14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.exe -o prog.c`

15. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`

16. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ le programme affiche x

18. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements

20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 8

☐ 16

☐ 4

☐ 0

2. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

☐ tous ensemble

☐ chacun son tour, après que le processus précédent a terminé

☐ tour à tour, un petit peu à chaque fois

☐ en parallèle, chacun dans un registre

3. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

☐ le terminal affiche `5`

☐ le terminal affiche `x = 2`

☐ le terminal affiche "Faux"

☐ le terminal affiche `x = 5`

4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

☐ retourner un bloc

☐ répéter un bloc tant qu'une condition est vérifée

☐ sélectionner entre deux blocs à l'aide d'une condition

☐ mettre les blocs en séquence les uns à la suite des autres

5. Une *segmentation fault* est une erreur qui survient lorsque :

☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur

☐ la division du programme en zones homogènes échoue

☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

☐ `Mineur`

☐ `Majeur`

☐ `Majeur Mineur`

☐ rien

7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1

☐ 0 1 2 3 4

☐ 1 2 3 4

☐ 4 3 2 1 0

8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

☐ la case mémoire 8 contiendra 16

☐ le terminal affiche 8

☐ le bus explose

☐ la case mémoire 8 contiendra 0

9. Dans la commande gcc, l'option `-Wall` signifie :

☐ qu'il faut lancer un déboggueur

☐ qu'on veut changer alétoirement de fond d'écran

☐ qu'il faut indenter le fichier source

☐ que l'on veut voir tous les avertissements

10. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Mineur`

☐ rien

☐ `Majeur`

☐ `Majeur Mineur`

11. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
12. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
13. Un registre du processeur est :
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.exe -Wall -o prog.c`
15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `1 2 1 2 3`
    - ☐ `0 0 0 1 1 1`
    - ☐ `0 1 0 1 0 1 0 1`
16. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0
18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 2 0 1 2 3`
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 3 0 1 2`
19. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ changer de répertoire courant
    - ☐ jouer de la musique
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ ouvir un bureau partagé (common desktop)
20. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

3. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 6
- ☐ 1
- ☐ 42

4. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ composé
   - ☐ lu, écrit, parlé

5. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

6. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
   - ☐ voir des clips musicaux

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16

8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2

9. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int k;`
   - ☐ `loop i;`
   - ☐ `int loop n;`
   - ☐ `int %d;`

10. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

12. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 16
- ☐ 3
- ☐ 6
- ☐ 20

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
- ☐ `char tableau[5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[taille=5];`
- ☐ `int toto[5];`
- ☐ `int[] new tableau(5);`

15. Un bit est :
- ☐ un battement d'horloge processeur
- ☐ l'instruction qui met fin à un programme
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)

16. Le bus système sert à :
- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur
- ☐ Transférer des données et intructions entre processeur et mémoire

17. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
- ☐ un point-virgule manquant
- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ une accolade manquante

18. Un fichier source est :
- ☐ un document de référence du système
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé

19. Quel est l'opérateur de différence en C :
- ☐ `!=`
- ☐ $\neq$
- ☐ `!`
- ☐ `<>`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
2. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule manquant
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
3. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
4. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
5. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 0
   - ☐ 15
   - ☐ 6
   - ☐ 10
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
9. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme
    - ☐ afficher le contenu d'un fichier texte
11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ 0 1 2 0 1 2
- ☐ 1 2 3 1 2
- ☐ 0 0 1 1 2 2
- ☐ 0 1 0 1 0 1

12. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

13. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `yppasswd`
- ☐ `mkdir TP4`
- ☐ `kwrite TP4`
- ☐ `new TP4`

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`

16. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ begin
- ☐ include
- ☐ main

17. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

18. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 16
- ☐ 20
- ☐ 3

19. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une directive préprocesseur `#include` manquante
- ☐ un caractère interdit en C
- ☐ une variable non déclarée
- ☐ une faute de frappe dans un appel de fonction

20. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ ouvrir un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un fichier texte
- ☐ créer un répertoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 2`

2. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1

3. Un bit est :
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire

4. Le langage C est un langage
   - ☐ compilé
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ composé

5. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Écrire des données sur le dique dur
   - ☐ Arriver à l'heure en cours
   - ☐ transporter les processus du tourniquet au processeur

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ en temps d'accès
   - ☐ certaines données de la mémoire de travail

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5

8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`

9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur

10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`

11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`

13. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 16
- ☐ 6
- ☐ 20
- ☐ 3

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = 4`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

16. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tour à tour, un petit peu à chaque fois
- ☐ en parallèle, chacun dans un registre
- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé

18. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur

19. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6

20. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ `<>`
- ☐ $\neq$
- ☐ `!=`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
3. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
   - ☐ ouvrir un fichier texte
4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x
   - ☐ le programme affiche `****`
5. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 42
   - ☐ 6
   - ☐ 1
   - ☐ 0
7. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)
8. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
10. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 1 2 3 4
11. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
12. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6

13. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier

14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ rien

15. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 8
    - ☐ 4
    - ☐ 0
    - ☐ 16

16. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `1 2 3 1 2`
    - ☐ `0 1 2 0 1 2`

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ en parallèle, chacun dans un registre
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ tous ensemble

18. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une variable non déclarée
    - ☐ une faute de frappe dans un appel de fonction

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`

20. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
2. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"
4. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ changer de répertoire courant
5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 8 2
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`
   - ☐ `#define N 3`
8. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une variable non déclarée
9. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ begin
   - ☐ init
   - ☐ main
10. Le langage C est un langage
    - ☐ interprété
    - ☐ composé
    - ☐ lu, écrit, parlé
    - ☐ compilé
11. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ rien
12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 4
    - ☐ 0
    - ☐ 16
    - ☐ 8

13. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6

14. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = %d`

16. Le bus système sert à :

- ☐ transporter les processus du tourniquet au processeur
- ☐ Arriver à l'heure en cours
- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Écrire des données sur le dique dur

17. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ un point-virgule manquant
- ☐ une accolade en trop
- ☐ une accolade manquante

18. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue

19. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

- ☐ `for(i=0;i<5;i=i+1)`
- ☐ `for(i=0;i<=5;i=i+1)`
- ☐ `for(i=1;i<5;i=i+1)`
- ☐ `for(i=1;i<=5;i=i+1)`

20. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le langage C est un langage
   - ☐ composé
   - ☐ interprété
   - ☐ compilé
   - ☐ lu, écrit, parlé
2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche ****
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
3. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
4. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 10
   - ☐ 15
   - ☐ 0
   - ☐ 6
5. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
6. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
7. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8
8. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
10. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 0 1 0 1`
    - ☐ `1 2 3 1 2`
11. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ répéter un bloc tant qu'une condition est vérifée

12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0
- ☐ le bus explose
- ☐ le terminal affiche 8

13. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade en trop
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
    - ☐ un point-virgule en trop

14. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ begin
    - ☐ init
    - ☐ main

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`
- ☐ `1 2 1 2 3`

16. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ compiler un programme

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`

18. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `<>`
    - ☐ $\neq$
    - ☐ `!`

19. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

20. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 3

3. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`

5. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 8
   - ☐ 0
   - ☐ 4

6. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ 0 0 1 1 2 2 3
   - ☐ 0 1 2 0 1 2 3
   - ☐ 0 1 2 0 1 2
   - ☐ 0 1 2 3 0 1 2

7. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7

8. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4

9. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
   - ☐ le terminal affiche 8

10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`

11. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur

12. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`

13. Quel est l'opérateur de différence en C :
    - ☐ `<>`
    - ☐ `!=`
    - ☐ `!`
    - ☐ $\neq$

14. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une variable non déclarée
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ un caractère interdit en C
    - ☐ une directive préprocesseur `#include` manquante

15. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`

16. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade manquante
    - ☐ une accolade en trop

17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ en parallèle, chacun dans un registre
    - ☐ tous ensemble

18. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ des processus
    - ☐ les fichiers du disque
    - ☐ en temps d'accès

19. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ la division du programme en zones homogènes échoue

20. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `yppasswd`
    - ☐ `mkdir TP4`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ begin
   - ☐ main
   - ☐ include
2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur Mineur`
3. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse sémantique
   - ☐ analyse harmonique
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
4. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ rien
5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
6. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
8. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ ouvrir un fichier texte
   - ☐ créer un répertoire
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
9. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une variable non déclarée
10. Le bus système sert à :
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Écrire des données sur le dique dur
11. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = 0`
- ☐ `j = 4`
- ☐ `j = %d`

12. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
13. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
14. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
15. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

16. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2 0
- ☐ 8 2

18. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ en temps d'accès
    - ☐ des processus
    - ☐ les fichiers du disque
19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
2. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 0 0 1 1 1`
4. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 42
   - ☐ 1
   - ☐ 6
   - ☐ 0
5. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`
7. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 0
   - ☐ 8
   - ☐ 4
8. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
9. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 4
   - ☐ 0
   - ☐ 16
   - ☐ 8
10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :

- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`

11. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
13. Le langage C est un langage
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé
    - ☐ compilé
14. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 5`
15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 0 1 0 1`
    - ☐ `1 2 3 1 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 1 1 2 2`
16. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
17. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
18. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int toto[taille=5];`
2. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
3. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 20
   - ☐ 6
   - ☐ 3
   - ☐ 16
4. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
5. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
6. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
7. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
8. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
9. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 4
   - ☐ 0
   - ☐ 16
10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3

11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
12. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
13. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers` ≪ `prinft` ≫
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse sémantique
14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `loop i;`
15. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
16. Le langage C est un langage
    - ☐ composé
    - ☐ compilé
    - ☐ interprété
    - ☐ lu, écrit, parlé
17. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 16
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le terminal affiche 8
18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6
19. Quel est l'opérateur de différence en C :
    - ☐ `!=`
    - ☐ `!`
    - ☐ `<>`
    - ☐ ≠
20. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse lexicale
    - ☐ analyse harmonique
    - ☐ analyse syntaxique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$

2. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale
   - ☐ analyse harmonique

3. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
   - ☐ `char tableau[5];`

4. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail

5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 0
   - ☐ 42
   - ☐ 1
   - ☐ 6

6. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!=`
   - ☐ `!`
   - ☐ $\neq$

7. Un fichier source est :
   - ☐ un document illisible pour les humains
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

8. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`

10. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source

11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ rien

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`

13. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
14. Un registre du processeur est :
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur
    - ☐ une unité de calcul spécialisée de l'ordinateur
15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
16. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

    - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
    - ☐ comportent une erreur qui ne sera pas détectée
    - ☐ ne comportent aucune erreur
    - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
17. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

18. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = %d`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 16
   - ☐ 8
   - ☐ 4
3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ `Majeur`
4. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un fichier texte
   - ☐ créer un répertoire
   - ☐ ouvrir un fichier texte
5. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 3
   - ☐ 16
   - ☐ 20
   - ☐ 6
6. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 8 2
7. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse lexicale
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse harmonique
8. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
9. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
10. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 0 1 0 1`
    - ☐ `0 0 1 1 2 2`
    - ☐ `0 1 2 0 1 2`
    - ☐ `1 2 3 1 2`

12. Le bus système sert à :

- ☐ Transférer des données et intructions entre processeur et mémoire
- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur
- ☐ Écrire des données sur le dique dur

13. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers « prinft »`

- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres

17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `0 0 0 1 1 1`
- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`

18. Un fichier source est :

- ☐ un document qui doit être protégé
- ☐ un document illisible pour les humains
- ☐ un document de référence du système
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x y);`

20. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ init
- ☐ begin
- ☐ include

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un document illisible pour les humains
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`

3. Le langage C est un langage
   - ☐ interprété
   - ☐ compilé
   - ☐ lu, écrit, parlé
   - ☐ composé

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16

5. Si cette erreur apparaît à la compilation :
   `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant

6. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 6
   - ☐ 0
   - ☐ 1
   - ☐ 42

7. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 0 1 2`

8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable y vaut 5

10. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int[] new tableau(5);`

11. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ répéter un bloc tant qu'une condition est vérifée

12. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`

13. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 0
- ☐ 10
- ☐ 6

14. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ voir des clips musicaux

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

16. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

17. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ que l'on veut voir tous les avertissements

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`

19. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

20. Un bit est :

- ☐ l'instruction qui met fin à un programme
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ un battement d'horloge processeur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
3. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ main
   - ☐ init
   - ☐ include
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 5`
5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
6. Quel est l'opérateur de différence en C :
   - ☐ `<>`
   - ☐ `!`
   - ☐ `!=`
   - ☐ $\neq$
7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
8. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n",x y);`
9. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `loop i;`
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?
    - ☐ `j = 4`
    - ☐ `j = 5`
    - ☐ `j = 0`
    - ☐ `j = %d`
11. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1

12. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

13. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 2

14. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x
- ☐ le programme affiche `****`

16. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source
- ☐ qu'on veut changer alétoirement de fond d'écran

17. Un bit est :

- ☐ la longueur d'un mot mémoire
- ☐ l'instruction qui met fin à un programme
- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)

18. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue

19. Le langage C est un langage

- ☐ composé
- ☐ interprété
- ☐ lu, écrit, parlé
- ☐ compilé

20. Sous unix (ou linux), la commande `cd` permet de :

- ☐ détruire un fichier
- ☐ changer de répertoire courant
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ ouvir un bureau partagé (common desktop)
- ☐ jouer de la musique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ détruire un fichier
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique

2. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 2
   - ☐ 8 6 4 2
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0

3. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1

4. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` ≪ `prinft` ≫ que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`

6. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé

7. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ 0 1 2 3 0 1 2
   - ☐ 0 1 2 0 1 2 3
   - ☐ 0 1 2 0 1 2
   - ☐ 0 0 1 1 2 2 3

9. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4

10. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1

11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 0 1 0 1`

12. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ les fichiers du disque
    - ☐ en temps d'accès
    - ☐ des processus
13. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ un battement d'horloge processeur
    - ☐ l'instruction qui met fin à un programme
    - ☐ la longueur d'un mot mémoire
14. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
15. Quel est l'opérateur de différence en C :
    - ☐ `!`
    - ☐ `<>`
    - ☐ $\neq$
    - ☐ `!=`
16. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
    - ☐ transporter les processus du tourniquet au processeur
17. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ begin
    - ☐ init
    - ☐ main
    - ☐ include
18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 10
- ☐ 0
- ☐ 15

19. Le langage C est un langage
    - ☐ composé
    - ☐ compilé
    - ☐ lu, écrit, parlé
    - ☐ interprété
20. Un fichier source est :
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé
    - ☐ un document de référence du système

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6

2. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** *** **** *****`
- ☐ `** ** ** ** ** **`
- ☐ `***** **** *** ***`

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

4. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

5. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse syntaxique
- ☐ analyse sémantique
- ☐ analyse harmonique

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`

8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

9. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ `!=`
- ☐ $\neq$
- ☐ `<>`

10. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`

12. Sous unix (ou linux), la commande `ls` permet de :

- ☐ compiler un programme
- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ afficher le contenu d'un fichier texte

13. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 0
- ☐ 1
- ☐ 42

14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `loop i;`
- ☐ `int k;`
- ☐ `int %d;`

15. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ la longueur d'un mot mémoire
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ sélectionner entre deux blocs à l'aide d'une condition

17. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ ouvrir un fichier texte
- ☐ changer de répertoire courant
- ☐ créer un répertoire
- ☐ créer un fichier texte

18. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

19. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`
- ☐ `int toto[5];`
- ☐ `int tab[] = 5;`
- ☐ `int[] new tableau(5);`

20. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 4
- ☐ 16
- ☐ 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 4
   - ☐ 0
   - ☐ 8
3. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`
4. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
5. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
   - ☐ l'analyse des entrées clavier
6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche x
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$
7. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
8. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 2 4 6
9. Un fichier source est :
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document illisible pour les humains
10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `int k;`
    - ☐ `int %d;`
    - ☐ `loop i;`
11. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
12. L'ordonnancement par tourniquet permet :
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ d'afficher des ronds colorés à l'écran
13. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ créer un fichier texte

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `char tableau[5];`
- ☐ `int tab[] = 5;`
- ☐ `int toto[taille=5];`
- ☐ `int toto[5];`
- ☐ `int[] new tableau(5);`

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien

16. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1`

17. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2 0

18. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `** ** ** ** ** **`
- ☐ `**** **** **** ****`

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

20. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

2. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source

3. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`

4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 0 1 1 2 2`

5. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ voir des clips musicaux

6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6

7. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`

8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois

9. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`

12. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ la division du programme en zones homogènes échoue

13. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

14. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction

- ☐ `int toto[5];`
- ☐ `int tab[] = 5;`
- ☐ `int[] new tableau(5);`
- ☐ `int toto[taille=5];`
- ☐ `char tableau[5];`

15. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 0
- ☐ 15
- ☐ 6

17. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade manquante
- ☐ un point-virgule en trop
- ☐ une accolade en trop
- ☐ un point-virgule manquant

18. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%d et y=%d\n",x y);`

19. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

20. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

2. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble

3. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop

4. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ que l'on veut voir tous les avertissements

5. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
   - ☐ `kwrite TP4`
   - ☐ `new TP4`

6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `loop i;`
   - ☐ `int k;`
   - ☐ `int %d;`
   - ☐ `int loop n;`

7. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran

8. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ certains programmes sont de vrais plats de spaghetti

9. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ certaines données de la mémoire de travail
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ les fichiers du disque

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$

11. Le langage C est un langage
    - ☐ compilé
    - ☐ composé
    - ☐ interprété
    - ☐ lu, écrit, parlé

12. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 0 1 2 3`

14. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ la division du programme en zones homogènes échoue

15. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur Mineur`

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`
- ☐ rien

17. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 6
- ☐ 15
- ☐ 10

18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 0 1 1 1`

19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 8 6 4 2 0

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3
- ☐ la variable y vaut 5

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 1 2 3 4
☐ 4 3 2 1
☐ 4 3 2 1 0
☐ 0 1 2 3 4

2. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 16
☐ 0
☐ 8
☐ 4

3. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

☐ `gcc -Wall prog.c -o prog.exe`
☐ `gcc prog.c -o -Wall prog.exe`
☐ `gcc -Wall prog.exe -o prog.c`
☐ `gcc prog.exe -Wall -o prog.c`

4. L'ordonnancement par tourniquet permet :

☐ de ne pas perdre de temps avec la commutation de contexte
☐ de doubler la mémoire disponible
☐ d'entretenir l'illusion que les processus tournent en parallèle
☐ d'afficher des ronds colorés à l'écran

5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1
☐ 0 1 2 3 4
☐ 4 3 2 1 0
☐ 0 1 2 3

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ `Mineur`
☐ `Majeur`
☐ `Majeur Mineur`
☐ rien

7. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :

☐ `for(i=0;i<5;i=i+1)`
☐ `for(i=1;i<5;i=i+1)`
☐ `for(i=0;i<=5;i=i+1)`
☐ `for(i=1;i<=5;i=i+1)`

8. Le bus système sert à :

☐ Transférer des données et intructions entre processeur et mémoire
☐ Écrire des données sur le dique dur
☐ transporter les processus du tourniquet au processeur
☐ Arriver à l'heure en cours

9. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

☐ un caractère interdit en C
☐ une faute de frappe dans un appel de fonction
☐ une variable non déclarée
☐ une directive préprocesseur `#include` manquante

10. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

☐ analyse harmonique
☐ analyse sémantique
☐ analyse lexicale
☐ analyse syntaxique

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

☐ comportent une erreur qui sera détectée au cours de l'édition de lien
☐ ne comportent aucune erreur
☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
☐ comportent une erreur qui ne sera pas détectée

12. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 0
- ☐ 15
- ☐ 6

13. Un registre du processeur est :
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une gamme de fréquence de fonctionnement du processeur

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

15. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ une accolade en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade manquante

16. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `int toto[5];`

17. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

18. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8

19. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

20. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ détruire un fichier
   - ☐ jouer de la musique
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = 5`

3. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée

4. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ include
   - ☐ begin
   - ☐ main

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"

6. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur

7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`

8. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$

10. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

11. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 4 3 2 1

12. Un registre du processeur est :

- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ un composant qui contient la liste des fichiers du système

13. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

14. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0

15. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

17. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`

18. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

19. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source

20. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 0 2 4 6 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ include
   - ☐ begin
   - ☐ main
   - ☐ init

2. Les lignes
```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

3. Le code suivant :
```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```
   affichera :
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0

4. Le code suivant :
```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```
   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1

5. Après exécution jusqu'à la ligne 15 du programme C :
```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```
   - ☐ le programme affiche x
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ la variable x vaut $-\frac{1}{2}$

6. Le code suivant :
```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```
   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Mineur`

7. Le code suivant :
```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```
   affichera :
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

8. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ détruire un fichier
   - ☐ changer de répertoire courant
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ jouer de la musique

9. Après exécution jusqu'à la ligne 15 du programme C :
```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5

10. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)

11. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 1 2 3 4

12. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse lexicale
- ☐ analyse harmonique
- ☐ analyse syntaxique
- ☐ analyse sémantique

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`

14. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tour à tour, un petit peu à chaque fois
- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ en parallèle, chacun dans un registre

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`

17. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 4
- ☐ 8

18. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ changer de répertoire courant
- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte

19. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

20. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
`Undefined symbols :"_prinft"` ou
`référence indéfinie vers ≪ prinft ≫`

- ☐ l'analyse sémantique
- ☐ l'édition de liens
- ☐ l'analyse des entrées clavier
- ☐ l'analyse harmonique

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
2. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
3. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `loop i;`
   - ☐ `int %d;`
   - ☐ `int k;`
4. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse sémantique
   - ☐ analyse harmonique
5. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ main
   - ☐ init
   - ☐ include
6. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
8. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
9. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 0
   - ☐ 10
   - ☐ 6
   - ☐ 15
10. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ détruire un fichier
    - ☐ changer de répertoire courant
    - ☐ ouvir un bureau partagé (common desktop)
11. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1
12. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut lancer un déboggueur
    - ☐ qu'il faut indenter le fichier source
    - ☐ que l'on veut voir tous les avertissements
13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`

14. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition
    - ☐ retourner un bloc
17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

18. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 4
- ☐ 0
- ☐ 16
- ☐ 8

19. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :
- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8

20. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document illisible pour les humains
    - ☐ un document qui doit être protégé

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`

2. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ afficher le contenu d'un fichier texte
3. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
4. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`

5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 20
- ☐ 16
- ☐ 3
- ☐ 6

7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
9. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `loop i;`
10. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=1;i<5;i=i+1)`
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7

12. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 0
- ☐ 6
- ☐ 15
- ☐ 10

13. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur

14. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur

15. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 4
- ☐ 16

16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4

17. Quels calculs peut-on programmer en programmation structurée ?

- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

18. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`

20. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche x

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le bus système sert à :
   - ☐ Écrire des données sur le dique dur
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
2. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
3. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ `Majeur Mineur`
4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 2 0 1 2 3`
   - ☐ `0 1 2 3 0 1 2`
   - ☐ `0 0 1 1 2 2 3`
5. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `kwrite TP4`
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `yppasswd`
6. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0
   - ☐ le bus explose
7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
8. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
9. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 0 0 1 1 1`
10. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6 8
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6
11. Dans la commande gcc, l'option `-Wall` signifie :
    - ☐ qu'il faut lancer un déboggueur
    - ☐ que l'on veut voir tous les avertissements
    - ☐ qu'on veut changer alétoirement de fond d'écran
    - ☐ qu'il faut indenter le fichier source

12. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 3
- ☐ 20
- ☐ 16

13. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ $\neq$
- ☐ `!=`
- ☐ `!`

14. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ les fichiers du disque
- ☐ en temps d'accès

15. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"

17. Sous unix (ou linux), la commande `ls` permet de :

- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme
- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte

18. Un fichier source est :

- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un document qui doit être protégé
- ☐ un document de référence du système
- ☐ un document illisible pour les humains
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur

19. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define taille = 3`
- ☐ `#define N 3`

20. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ main
- ☐ begin
- ☐ include

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = N`
   - ☐ `#define taille = 3`
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
2. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 1 2 3`
   - ☐ `0 0 0 1 1 1`
4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
5. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ init
   - ☐ begin
   - ☐ main
   - ☐ include
6. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ compiler un programme
8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 0
   - ☐ 4
   - ☐ 8
   - ☐ 16
9. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 2
10. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5

12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

13. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 1
- ☐ 42
- ☐ 6

14. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail
- ☐ en temps d'accès

15. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc
- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition

16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n",x y);`

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ le programme affiche x

18. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'il faut indenter le fichier source

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 16
- ☐ 8
- ☐ 4

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = 4`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quel est l'opérateur de différence en C :
   - ☐ `!`
   - ☐ `<>`
   - ☐ `!=`
   - ☐ $\neq$
2. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ détruire un fichier
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 0`
4. Un fichier source est :
   - ☐ un document illisible pour les humains
   - ☐ un document qui doit être protégé
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document de référence du système
   - ☐ un fichier texte qui sera traduit en instructions processeur
5. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
6. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
8. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse harmonique
   - ☐ analyse sémantique
9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
10. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 4
    - ☐ 0
    - ☐ 8
    - ☐ 16
11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
12. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ `Mineur`
    - ☐ rien

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `1 2 3 1 2`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`

14. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`
- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`

15. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois
- ☐ tous ensemble

16. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 0 1 2 3

17. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`

20. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`

2. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 0
- ☐ 10
- ☐ 15

3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define N 3`
- ☐ `#define taille = 3`

4. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

5. Quel est l'opérateur de différence en C :

- ☐ $\neq$
- ☐ `!=`
- ☐ `!`
- ☐ `<>`

6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `int k;`
- ☐ `loop i;`
- ☐ `int %d;`

7. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 1 2 3 4 5 6

8. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`

9. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

10. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

11. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 0
- ☐ 4

12. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 0 1 1 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 0 1 0 1 0 1`
- ☐ `1 2 1 2 3`

15. Le langage C est un langage

- ☐ lu, écrit, parlé
- ☐ interprété
- ☐ composé
- ☐ compilé

16. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ include
- ☐ main
- ☐ begin
- ☐ init

17. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Mineur`
- ☐ `Majeur`

18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`

19. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade manquante
- ☐ un point-virgule manquant
- ☐ une accolade en trop
- ☐ un point-virgule en trop

20. L'ordonnancement par tourniquet permet :

- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'afficher des ronds colorés à l'écran
- ☐ d'entretenir l'illusion que les processus tournent en parallèle

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%x et y=%y\n");`
2. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2 0
   - ☐ 8 2
   - ☐ 8 6 4 2
3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ 1 2 3 1 2
   - ☐ 0 1 0 1 0 1
   - ☐ 0 0 1 1 2 2
   - ☐ 0 1 2 0 1 2
4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int k;`
   - ☐ `loop i;`
   - ☐ `int %d;`
5. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
6. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ changer de répertoire courant
   - ☐ créer un répertoire
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
7. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
8. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ 0 0 0 1 1 1
    - ☐ 0 1 2 0 1 2
    - ☐ 0 1 0 1 0 1 0 1
    - ☐ 1 2 1 2 3

11. L'ordonnancement par tourniquet permet :

- ☐ de ne pas perdre de temps avec la commutation de contexte
- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

13. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`

14. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ un point-virgule manquant
- ☐ une accolade manquante
- ☐ une accolade en trop

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 0 1 1 2 2 3`

16. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`

18. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

19. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 1 2 3 4

20. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tous ensemble
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ en parallèle, chacun dans un registre
- ☐ tour à tour, un petit peu à chaque fois

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = 5`

2. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail

3. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

qu'est ce qui sera affiché ?

- ☐ `**** **** **** ****`
- ☐ `** ** ** ** ** **`
- ☐ `** *** **** *****`
- ☐ `***** **** *** ***`

4. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int toto[5];`
   - ☐ `char tableau[5];`
   - ☐ `int toto[taille=5];`
   - ☐ `int tab[] = 5;`
   - ☐ `int[] new tableau(5);`

5. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

6. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`

7. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`

8. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'il faut indenter le fichier source
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran

9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois

10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`

11. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 16
- ☐ 0

13. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur

14. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 0
- ☐ 16

15. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher le contenu d'un fichier texte
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5

17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

18. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0

19. Quel est l'opérateur de différence en C :

- ☐ `!=`
- ☐ `!`
- ☐ `<>`
- ☐ ≠

20. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define taille = 3`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x

2. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
   - ☐ une directive préprocesseur `#include` manquante

3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 5`
   - ☐ `j = 0`

4. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6

5. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"

6. Le langage C est un langage
   - ☐ composé
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ compilé

7. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux

8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`

10. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`

11. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable x vaut 5 et la variable y vaut 0
    - ☐ la variable x vaut 0
    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"

13. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`

14. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

15. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `char tableau[5];`
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`

16. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 16
    - ☐ 8
    - ☐ 0
    - ☐ 4

17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N =  3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`

18. Un fichier source est :
    - ☐ un document illisible pour les humains
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
    - ☐ un document de référence du système
    - ☐ un fichier texte qui sera traduit en instructions processeur

19. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 0 1 2 3 4

20. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ détruire un fichier
    - ☐ changer de répertoire courant

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
2. Un registre du processeur est :
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une gamme de fréquence de fonctionnement du processeur
3. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 4`
   - ☐ `j = %d`
   - ☐ `j = 5`
   - ☐ `j = 0`
4. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
5. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4
6. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
7. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `5`
8. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 16
   - ☐ 4
   - ☐ 0
9. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 0
   - ☐ 15
   - ☐ 10
   - ☐ 6
10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ `Majeur`
    - ☐ rien

11. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 16
- ☐ 6
- ☐ 20
- ☐ 3

12. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 4
- ☐ 8
- ☐ 16

13. Sous unix (ou linux), la commande `cd` permet de :

- ☐ ouvir un bureau partagé (common desktop)
- ☐ jouer de la musique
- ☐ changer de répertoire courant
- ☐ détruire un fichier
- ☐ récupérer un programme arrêté avec la commande `ab`

14. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `kwrite TP4`
- ☐ `mkdir TP4`
- ☐ `yppasswd`
- ☐ `new TP4`

15. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 1 2 3 4
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1

17. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`

18. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse sémantique
- ☐ analyse syntaxique
- ☐ analyse lexicale
- ☐ analyse harmonique

19. Quel est l'opérateur de différence en C :

- ☐ `!`
- ☐ `<>`
- ☐ $\neq$
- ☐ `!=`

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
2. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
3. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ transporter les processus du tourniquet au processeur
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
4. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
6. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `char tableau[5];`
   - ☐ `int toto[5];`
   - ☐ `int[] new tableau(5);`
   - ☐ `int tab[] = 5;`
   - ☐ `int toto[taille=5];`
7. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ 0 0 1 1 2 2 3
   - ☐ 0 1 2 0 1 2 3
   - ☐ 0 1 2 3 0 1 2
   - ☐ 0 1 2 0 1 2
9. Si cette erreur apparaît à la compilation :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?
   - ☐ une variable non déclarée
   - ☐ un caractère interdit en C
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une faute de frappe dans un appel de fonction
10. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define N 3`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`
11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8
    - ☐ 0 2 4 6
12. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

13. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
14. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`
15. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
16. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `kwrite TP4`
    - ☐ `mkdir TP4`
    - ☐ `new TP4`
    - ☐ `yppasswd`
17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ la variable y vaut 5
    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0
18. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur`
    - ☐ `Majeur Mineur`
    - ☐ `Mineur`
    - ☐ rien
19. Un bit est :
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ la longueur d'un mot mémoire
    - ☐ l'instruction qui met fin à un programme
20. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Mineur`
    - ☐ `Majeur`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ des processus
   - ☐ en temps d'accès
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail
2. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
3. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
4. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien
5. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 16
   - ☐ 20
   - ☐ 3
   - ☐ 6
7. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ rien
   - ☐ `Mineur`
   - ☐ `Majeur`
8. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 0`
   - ☐ `j = 4`
9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studlib.h>`
10. L'ordonnancement par tourniquet permet :
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
11. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

12. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
    - ☐ `mkdir TP4`
    - ☐ `yppasswd`
    - ☐ `new TP4`
    - ☐ `kwrite TP4`
13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc -Wall prog.exe -o prog.c`
14. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ créer un fichier texte
    - ☐ créer un répertoire
    - ☐ changer de répertoire courant
15. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ une variable non déclarée
16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `int loop n;`
    - ☐ `loop i;`
    - ☐ `int k;`
17. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

18. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int tab[] = 5;`
    - ☐ `int[] new tableau(5);`
    - ☐ `int toto[5];`
    - ☐ `char tableau[5];`
    - ☐ `int toto[taille=5];`
19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studio.h>`
20. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ 0 1 0 1 0 1
- ☐ 0 1 2 0 1 2
- ☐ 1 2 3 1 2
- ☐ 0 0 1 1 2 2

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `new TP4`
   - ☐ `yppasswd`
2. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ que l'on veut voir tous les avertissements
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
   - ☐ qu'on veut changer alétoirement de fond d'écran
3. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2
   - ☐ 8 6 4 2 0
   - ☐ 8 2
   - ☐ 0 2 4 6 8
4. Le langage C est un langage
   - ☐ interprété
   - ☐ composé
   - ☐ lu, écrit, parlé
   - ☐ compilé
5. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define N =  3`
   - ☐ `#define taille = 3`
6. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
7. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
   - ☐ en parallèle, chacun dans un registre
   - ☐ tour à tour, un petit peu à chaque fois
8. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur
9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable y vaut 5
   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
10. Quels calculs peut-on programmer en programmation structurée ?
    - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
    - ☐ certains programmes sont de vrais plats de spaghetti
    - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
    - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
11. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ en temps d'accès
    - ☐ certaines données de la mémoire de travail
    - ☐ des processus
    - ☐ les fichiers du disque
12. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 1 2 3 4
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 4 3 2 1
13. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

14. L'ordonnancement par tourniquet permet :
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de doubler la mémoire disponible
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ 0 1 2 0 1 2 3
- ☐ 0 1 2 3 0 1 2
- ☐ 0 1 2 0 1 2
- ☐ 0 0 1 1 2 2 3

16. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
    - ☐ retourner un bloc
    - ☐ mettre les blocs en séquence les uns à la suite des autres
    - ☐ répéter un bloc tant qu'une condition est vérifée
    - ☐ sélectionner entre deux blocs à l'aide d'une condition

17. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ le programme affiche x
    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche ****
    - ☐ la variable x vaut 16

18. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 10
- ☐ 0
- ☐ 6
- ☐ 15

19. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 0
- ☐ 4
- ☐ 16

20. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document qui doit être protégé
   - ☐ un document de référence du système
   - ☐ un document illisible pour les humains
   - ☐ un fichier texte qui sera traduit en instructions processeur

2. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 0 1 1 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `0 1 0 1 0 1 0 1`
   - ☐ `1 2 1 2 3`

3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define N =  3`
   - ☐ `#define taille = N`
   - ☐ `#define N 3`
   - ☐ `#define taille = 3`

4. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x

5. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <stdio.h>`

6. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ compilé
   - ☐ composé

7. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `***** **** *** ***`
   - ☐ `**** **** **** ****`
   - ☐ `** ** ** ** ** **`
   - ☐ `** *** **** *****`

8. Un registre du processeur est :
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une unité de calcul spécialisée de l'ordinateur
   - ☐ un composant qui contient la liste des fichiers du système

9. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ tour à tour, un petit peu à chaque fois

10. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `5`
    - ☐ le terminal affiche "Faux"

11. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8

12. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `kwrite TP4`
- ☐ `mkdir TP4`

13. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

14. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

15. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <stdio.h>`

16. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `int loop n;`
- ☐ `int %d;`
- ☐ `loop i;`
- ☐ `int k;`

17. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ la case mémoire 8 contiendra 0
- ☐ le terminal affiche 8
- ☐ le bus explose

18. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ begin
- ☐ include
- ☐ init

19. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ une accolade manquante
- ☐ une accolade en trop
- ☐ un point-virgule en trop
- ☐ un point-virgule manquant

20. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 15
- ☐ 0
- ☐ 10

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tous ensemble
   - ☐ tour à tour, un petit peu à chaque fois

2. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ certains programmes sont de vrais plats de spaghetti
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine

3. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur`
   - ☐ `Majeur Mineur`

4. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 4
   - ☐ 0
   - ☐ 16

5. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ begin
   - ☐ include
   - ☐ init

6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ `Mineur`
   - ☐ `Majeur`
   - ☐ rien
   - ☐ `Majeur Mineur`

7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0

8. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`

9. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6 7
   - ☐ 0 2 4 6 8
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6

10. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable y vaut 5
    - ☐ la variable x vaut 0
    - ☐ la variable x vaut 5 et la variable y vaut 0

11. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

    ☐ `loop i;`

    ☐ `int k;`

    ☐ `int loop n;`

    ☐ `int %d;`

12. Le langage C est un langage

    ☐ compilé

    ☐ composé

    ☐ interprété

    ☐ lu, écrit, parlé

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?

    ☐ `0 0 1 1 2 2 3`

    ☐ `0 1 2 0 1 2`

    ☐ `0 1 2 3 0 1 2`

    ☐ `0 1 2 0 1 2 3`

14. Si cette erreur apparaît à la compilation :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫` que doit-on chercher dans le programme ?

    ☐ une faute de frappe dans un appel de fonction

    ☐ un caractère interdit en C

    ☐ une variable non déclarée

    ☐ une directive préprocesseur `#include` manquante

15. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`

    ☐ l'édition de liens

    ☐ l'analyse des entrées clavier

    ☐ l'analyse harmonique

    ☐ l'analyse sémantique

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

    ☐ `gcc prog.c -o -Wall prog.exe`

    ☐ `gcc -Wall prog.c -o prog.exe`

    ☐ `gcc -Wall prog.exe -o prog.c`

    ☐ `gcc prog.exe -Wall -o prog.c`

17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?

    ☐ `j = %d`

    ☐ `j = 4`

    ☐ `j = 0`

    ☐ `j = 5`

18. Le bus système sert à :

    ☐ Transférer des données et intructions entre processeur et mémoire

    ☐ transporter les processus du tourniquet au processeur

    ☐ Écrire des données sur le dique dur

    ☐ Arriver à l'heure en cours

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

    ☐ `#include <studlib.h>`

    ☐ `#appart <stdlib.h>`

    ☐ `#include <stdio.h>`

    ☐ `#include <studio.h>`

20. Dans la commande gcc, l'option `-Wall` signifie :

    ☐ que l'on veut voir tous les avertissements

    ☐ qu'on veut changer alétoirement de fond d'écran

    ☐ qu'il faut indenter le fichier source

    ☐ qu'il faut lancer un déboggueur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche `****`
   - ☐ la variable x vaut 16
   - ☐ le programme affiche x

2. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ composé
   - ☐ interprété

3. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ compiler un programme
   - ☐ afficher le contenu d'un fichier texte

4. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int loop n;`
   - ☐ `int k;`
   - ☐ `loop i;`
   - ☐ `int %d;`

5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Mineur`
   - ☐ rien
   - ☐ `Majeur`

6. L'ordonnancement par tourniquet permet :
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`

8. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse syntaxique
   - ☐ analyse lexicale
   - ☐ analyse sémantique
   - ☐ analyse harmonique

9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`
   - ☐ `1 2 3 1 2`

10. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 0 1 2 3 4

11. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4

12. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 0`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 4`

13. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 0
- ☐ 8
- ☐ 16
- ☐ 4

14. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = 5`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 4`

15. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ certaines données de la mémoire de travail
- ☐ des processus
- ☐ les fichiers du disque

16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 0
- ☐ 6
- ☐ 10

17. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée

18. Un fichier source est :

- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
- ☐ un document illisible pour les humains
- ☐ un document de référence du système
- ☐ un document qui doit être protégé
- ☐ un fichier texte qui sera traduit en instructions processeur

19. Quel est l'opérateur de différence en C :

- ☐ `<>`
- ☐ `!`
- ☐ $\neq$
- ☐ `!=`

20. Un bit est :

- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)
- ☐ l'instruction qui met fin à un programme
- ☐ la longueur d'un mot mémoire

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ tour à tour, un petit peu à chaque fois
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ en parallèle, chacun dans un registre

2. Un bit est :
   - ☐ la longueur d'un mot mémoire
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ l'instruction qui met fin à un programme
   - ☐ un battement d'horloge processeur

3. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 0

4. Quel est l'opérateur de différence en C :
   - ☐ $\neq$
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`

5. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 8 6 4 2 0
   - ☐ 8 6 4 2
   - ☐ 8 2
   - ☐ 0 2 4 6 8

6. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3 4 5 6
   - ☐ 0 2 4 6 8
   - ☐ 0 2 4 6
   - ☐ 0 1 2 3 4 5 6 7

7. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0

8. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ jouer de la musique
   - ☐ changer de répertoire courant
   - ☐ détruire un fichier

9. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ un point-virgule en trop
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop

10. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

    La valeur affichée est :
    - ☐ 0
    - ☐ 16
    - ☐ 4
    - ☐ 8

11. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

    - ☐ le programme affiche "Faux"
    - ☐ la variable x vaut 3
    - ☐ la variable x vaut 5 et la variable y vaut 3
    - ☐ la variable y vaut 5

12. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
    - ☐ `for(i=0;i<5;i=i+1)`
    - ☐ `for(i=0;i<=5;i=i+1)`
    - ☐ `for(i=1;i<=5;i=i+1)`
    - ☐ `for(i=1;i<5;i=i+1)`
13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3 4
    - ☐ 1 2 3 4
14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
15. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse sémantique
    - ☐ l'édition de liens
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier
16. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 0
    - ☐ 6
    - ☐ 42
17. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 4 3 2 1 0
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`
19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", i);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 0 1 0 1 0 1`
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 0 0 1 1 1`
    - ☐ `1 2 1 2 3`
20. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 3
    - ☐ 20
    - ☐ 6
    - ☐ 16

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse sémantique
   - ☐ analyse syntaxique
   - ☐ analyse lexicale

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5
   - ☐ le programme affiche "Faux"

3. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

   - ☐ le bus explose
   - ☐ la case mémoire 8 contiendra 16
   - ☐ le terminal affiche 8
   - ☐ la case mémoire 8 contiendra 0

4. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche "Faux"

5. Le langage C est un langage
   - ☐ composé
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ interprété

6. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

7. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 0 2 4 6 8
   - ☐ 8 6 4 2
   - ☐ 8 2
   - ☐ 8 6 4 2 0

8. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
   - ☐ `int %d;`
   - ☐ `int k;`
   - ☐ `int loop n;`
   - ☐ `loop i;`

9. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ begin
   - ☐ main
   - ☐ include
   - ☐ init

10. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ détruire un fichier

11. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

☐ le programme affiche ****

☐ la variable x vaut $-\frac{1}{2}$

☐ le programme affiche x

☐ la variable x vaut 16

13. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 4 3 2 1 0

☐ 4 3 2 1

☐ 0 1 2 3

☐ 0 1 2 3 4

14. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 0 2 4 6

☐ 0 1 2 3 4 5 6

☐ 0 1 2 3 4 5 6 7

☐ 0 2 4 6 8

15. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

☐ 16

☐ 8

☐ 0

☐ 4

16. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

☐ rien

☐ `Majeur`

☐ `Majeur Mineur`

☐ `Mineur`

17. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

☐ `j = 0`

☐ `j = %d`

☐ `j = 4`

☐ `j = 5`

18. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

☐ 1 2 3 4

☐ 4 3 2 1 0

☐ 0 1 2 3 4

☐ 4 3 2 1

19. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

☐ `0 1 2 0 1 2 3`

☐ `0 1 2 0 1 2`

☐ `0 1 2 3 0 1 2`

☐ `0 0 1 1 2 2 3`

20. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

☐ une accolade manquante

☐ un point-virgule en trop

☐ une accolade en trop

☐ un point-virgule manquant

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

   - ☐ la variable x vaut 0
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 5 et la variable y vaut 0
   - ☐ la variable y vaut 5

2. Le bus système sert à :
   - ☐ Transférer des données et intructions entre processeur et mémoire
   - ☐ Arriver à l'heure en cours
   - ☐ Écrire des données sur le dique dur
   - ☐ transporter les processus du tourniquet au processeur

3. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 1 2 3 4
   - ☐ 0 1 2 3 4

4. Quels calculs peut-on programmer en programmation structurée ?
   - ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
   - ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
   - ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine
   - ☐ certains programmes sont de vrais plats de spaghetti

5. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

   affichera :
   - ☐ 42
   - ☐ 6
   - ☐ 0
   - ☐ 1

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ le programme affiche ****
   - ☐ la variable x vaut $-\frac{1}{2}$
   - ☐ le programme affiche x
   - ☐ la variable x vaut 16

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable y vaut 5

8. Le langage C est un langage
   - ☐ interprété
   - ☐ composé
   - ☐ lu, écrit, parlé
   - ☐ compilé

9. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 3 1 2`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 1 2 0 1 2`

10. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier
    - ☐ récupérer un programme arrêté avec la commande `ab`

11. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
    - ☐ `Mineur`

12. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
    - ☐ créer un fichier texte

13. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 16
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 0
    - ☐ le terminal affiche 8

14. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ init
    - ☐ begin
    - ☐ main
    - ☐ include

15. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 1 2 3 4
    - ☐ 0 1 2 3
    - ☐ 4 3 2 1
    - ☐ 4 3 2 1 0

16. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers « prinft »`
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens

17. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

    La valeur de somme affichée est :
    - ☐ 20
    - ☐ 6
    - ☐ 3
    - ☐ 16

18. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

    - ☐ le terminal affiche "Faux"
    - ☐ le terminal affiche `x = 2`
    - ☐ le terminal affiche `x = 5`
    - ☐ le terminal affiche `5`

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x y);`

20. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
   - ☐ une faute de frappe dans un appel de fonction
2. Pour afficher à l'aide de `printf("%d\n",tab[i]);` le contenu d'un tableau de 5 entiers initialisé au préalable, on utilise plutôt :
   - ☐ `for(i=1;i<=5;i=i+1)`
   - ☐ `for(i=0;i<5;i=i+1)`
   - ☐ `for(i=0;i<=5;i=i+1)`
   - ☐ `for(i=1;i<5;i=i+1)`
3. Dans la commande gcc, l'option `-Wall` signifie :
   - ☐ qu'on veut changer alétoirement de fond d'écran
   - ☐ qu'il faut indenter le fichier source
   - ☐ qu'il faut lancer un déboggueur
   - ☐ que l'on veut voir tous les avertissements
4. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »`
   - ☐ l'analyse sémantique
   - ☐ l'analyse des entrées clavier
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
5. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ `Mineur`

6. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x

7. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6

8. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ la variable x vaut 0
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 5 et la variable y vaut 0
- ☐ la variable y vaut 5

9. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 16
- ☐ 3
- ☐ 20

10. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ changer de répertoire courant
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
    - ☐ jouer de la musique
11. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ voir des clips musicaux
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ afficher la liste de fichiers contenus dans un répertoire
12. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%x et y=%y\n");`

13. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 16
- ☐ 0

14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.exe -Wall -o prog.c`

15. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`
- ☐ `Majeur Mineur`

16. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4

17. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = N`
- ☐ `#define N 3`
- ☐ `#define taille = 3`
- ☐ `#define N =  3`

18. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs

19. Quels calculs peut-on programmer en programmation structurée ?

- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

20. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

2. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3
- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0

4. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`

5. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?

- ☐ une directive préprocesseur `#include` manquante
- ☐ une variable non déclarée
- ☐ un caractère interdit en C
- ☐ une faute de frappe dans un appel de fonction

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ certaines données de la mémoire de travail
- ☐ les fichiers du disque
- ☐ des processus
- ☐ en temps d'accès

7. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16

8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique

9. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studlib.h>`
- ☐ `#appart <stdlib.h>`
- ☐ `#include <studio.h>`
- ☐ `#include <stdio.h>`

10. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :

- ☐ `new TP4`
- ☐ `yppasswd`
- ☐ `mkdir TP4`
- ☐ `kwrite TP4`

11. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2

12. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un répertoire
- ☐ ouvrir un fichier texte
- ☐ créer un fichier texte
- ☐ changer de répertoire courant

13. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`

14. Le bus système sert à :

- ☐ Écrire des données sur le dique dur
- ☐ Arriver à l'heure en cours
- ☐ transporter les processus du tourniquet au processeur
- ☐ Transférer des données et intructions entre processeur et mémoire

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 6
- ☐ 1
- ☐ 42
- ☐ 0

16. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 0 1 2 3
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1

17. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6
- ☐ 0 1 2 3 4 5 6
- ☐ 0 2 4 6 8

18. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 8
- ☐ 0
- ☐ 4

19. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

- ☐ répéter un bloc tant qu'une condition est vérifée
- ☐ sélectionner entre deux blocs à l'aide d'une condition
- ☐ mettre les blocs en séquence les uns à la suite des autres
- ☐ retourner un bloc

20. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`

2. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur`
- ☐ rien
- ☐ `Majeur Mineur`
- ☐ `Mineur`

3. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 0 1 2 3

4. Un registre du processeur est :

- ☐ un composant qui contient la liste des fichiers du système
- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur

5. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ ne comportent aucune erreur
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui ne sera pas détectée

6. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

- ☐ `loop i;`
- ☐ `int %d;`
- ☐ `int loop n;`
- ☐ `int k;`

7. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ des processus
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail

8. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le bus explose
- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16

9. L'ordonnancement par tourniquet permet :

- ☐ d'entretenir l'illusion que les processus tournent en parallèle
- ☐ d'afficher des ronds colorés à l'écran
- ☐ de doubler la mémoire disponible
- ☐ de ne pas perdre de temps avec la commutation de contexte

10. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 16
- ☐ 3
- ☐ 20

11. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ que l'on veut voir tous les avertissements
- ☐ qu'on veut changer alétoirement de fond d'écran

12. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ main
- ☐ include
- ☐ begin
- ☐ init

13. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 6 4 2

14. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ créer un répertoire
- ☐ changer de répertoire courant
- ☐ ouvrir un fichier texte

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 1
- ☐ 0
- ☐ 42
- ☐ 6

16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5

17. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 16
- ☐ 0
- ☐ 8
- ☐ 4

18. Sous unix (ou linux), la commande `cd` permet de :

- ☐ ouvir un bureau partagé (common desktop)
- ☐ récupérer un programme arrêté avec la commande `ab`
- ☐ changer de répertoire courant
- ☐ détruire un fichier
- ☐ jouer de la musique

19. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x,y);`
- ☐ `printf("x=%x et y=%y\n");`

20. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un document illisible pour les humains
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
2. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc -Wall prog.c -o prog.exe`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.exe -o prog.c`
3. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
   - ☐ `#define taille = 3`
   - ☐ `#define taille = N`
   - ☐ `#define N =  3`
   - ☐ `#define N 3`
4. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

   qu'est ce qui sera affiché par ce printf ?
   - ☐ `j = 5`
   - ☐ `j = %d`
   - ☐ `j = 4`
   - ☐ `j = 0`
5. Un bit est :
   - ☐ l'instruction qui met fin à un programme
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ un battement d'horloge processeur
   - ☐ la longueur d'un mot mémoire
6. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
7. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

   qu'est ce qui sera affiché ?
   - ☐ `j = 4`
   - ☐ `j = 0`
   - ☐ `j = 5`
   - ☐ `j = %d`
8. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'édition de liens
   - ☐ l'analyse harmonique
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse sémantique
9. Un registre du processeur est :
   - ☐ un composant qui contient la liste des fichiers du système
   - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
   - ☐ une gamme de fréquence de fonctionnement du processeur
   - ☐ une unité de calcul spécialisée de l'ordinateur
10. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

    affichera :
    - ☐ `Mineur`
    - ☐ `Majeur Mineur`
    - ☐ rien
    - ☐ `Majeur`
11. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 8 2
    - ☐ 8 6 4 2
    - ☐ 8 6 4 2 0
    - ☐ 0 2 4 6 8

12. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :

- ☐ tour à tour, un petit peu à chaque fois
- ☐ chacun son tour, après que le processus précédent a terminé
- ☐ tous ensemble
- ☐ en parallèle, chacun dans un registre

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y;
13
14      y = x;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 0
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 0

14. Dans la commande gcc, l'option `-Wall` signifie :

- ☐ qu'il faut indenter le fichier source
- ☐ qu'il faut lancer un déboggueur
- ☐ qu'on veut changer alétoirement de fond d'écran
- ☐ que l'on veut voir tous les avertissements

15. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 42
- ☐ 6
- ☐ 1
- ☐ 0

16. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4

17. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ init
- ☐ include
- ☐ main
- ☐ begin

18. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 2 4 6 8
- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6

19. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 4
- ☐ 8
- ☐ 16
- ☐ 0

20. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ le terminal affiche 8
- ☐ la case mémoire 8 contiendra 0
- ☐ la case mémoire 8 contiendra 16
- ☐ le bus explose

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <stdio.h>`
2. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ de doubler la mémoire disponible
   - ☐ d'afficher des ronds colorés à l'écran
3. Si cette erreur apparaît à la compilation : `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
   - ☐ une accolade manquante
   - ☐ un point-virgule manquant
   - ☐ une accolade en trop
   - ☐ un point-virgule en trop
4. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
   - ☐ `printf("x=%d et y=%d\n",x,y);`
   - ☐ `printf("x=%d et y=%d\n,x,y");`
   - ☐ `printf("x=%d et y=%d\n",x y);`
   - ☐ `printf("x=%x et y=%y\n");`
5. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ compilé
   - ☐ composé
6. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

   affichera :
   - ☐ rien
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
7. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ un caractère interdit en C
   - ☐ une variable non déclarée
8. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
   - ☐ en parallèle, chacun dans un registre
   - ☐ tous ensemble
   - ☐ chacun son tour, après que le processus précédent a terminé
   - ☐ tour à tour, un petit peu à chaque fois
9. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
   - ☐ ne comportent aucune erreur
10. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ la case mémoire 8 contiendra 0
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 16
    - ☐ le terminal affiche 8
11. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Arriver à l'heure en cours
    - ☐ Écrire des données sur le dique dur
12. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

    affichera :
    - ☐ 0 2 4 6
    - ☐ 0 1 2 3 4 5 6
    - ☐ 0 1 2 3 4 5 6 7
    - ☐ 0 2 4 6 8

13. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `1 2 3 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`
- ☐ `0 1 2 0 1 2`

14. Quel est l'opérateur de différence en C :

- ☐ $\neq$
- ☐ `!`
- ☐ `!=`
- ☐ `<>`

15. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

16. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `x = 5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `5`

17. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une gamme de fréquence de fonctionnement du processeur
- ☐ une unité de calcul spécialisée de l'ordinateur

18. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 4
- ☐ 0

19. Une *segmentation fault* est une erreur qui survient lorsque :

- ☐ la division du programme en zones homogènes échoue
- ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
- ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
- ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

20. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ main
- ☐ include
- ☐ init

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Quel est l'opérateur de différence en C :
   - ☐ `!=`
   - ☐ `!`
   - ☐ `<>`
   - ☐ $\neq$

2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

   - ☐ le programme affiche "Faux"
   - ☐ la variable x vaut 3
   - ☐ la variable x vaut 5 et la variable y vaut 3
   - ☐ la variable y vaut 5

3. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire

4. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
   - ☐ une faute de frappe dans un appel de fonction
   - ☐ une directive préprocesseur `#include` manquante
   - ☐ une variable non déclarée
   - ☐ un caractère interdit en C

5. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

6. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 8
   - ☐ 16
   - ☐ 0
   - ☐ 4

7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <stdio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#include <studlib.h>`

8. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

   - ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
   - ☐ ne comportent aucune erreur
   - ☐ comportent une erreur qui ne sera pas détectée
   - ☐ comportent une erreur qui sera détectée au cours de l'édition de lien

9. Le langage C est un langage
   - ☐ interprété
   - ☐ lu, écrit, parlé
   - ☐ compilé
   - ☐ composé

10. L'ordonnancement par tourniquet permet :
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ de doubler la mémoire disponible

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#include <studio.h>`

12. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%x et y=%y\n");`

13. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1 0
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1

14. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un répertoire
    - ☐ changer de répertoire courant
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
15. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int k;`
    - ☐ `int loop n;`
    - ☐ `loop i;`
    - ☐ `int %d;`
16. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse sémantique
    - ☐ l'analyse harmonique
    - ☐ l'analyse des entrées clavier
    - ☐ l'édition de liens
17. Sous unix (ou linux), la commande `cd` permet de :
    - ☐ jouer de la musique
    - ☐ changer de répertoire courant
    - ☐ récupérer un programme arrêté avec la commande `ab`
    - ☐ détruire un fichier
    - ☐ ouvir un bureau partagé (common desktop)
18. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 4
- ☐ 16
- ☐ 0

19. Le bus système sert à :
    - ☐ Transférer des données et intructions entre processeur et mémoire
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur
    - ☐ Arriver à l'heure en cours
20. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
    - ☐ `#define N =  3`
    - ☐ `#define taille = N`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2 0
- ☐ 8 2
- ☐ 8 6 4 2
- ☐ 0 2 4 6 8

2. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

affichera :

- ☐ 0
- ☐ 42
- ☐ 1
- ☐ 6

3. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

4. Un programme en langage C doit comporter une et une seule définition de la fonction :

- ☐ begin
- ☐ include
- ☐ main
- ☐ init

5. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ le programme affiche x
- ☐ le programme affiche `****`
- ☐ la variable x vaut 16
- ☐ la variable x vaut $-\frac{1}{2}$

6. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 20
- ☐ 6
- ☐ 16
- ☐ 3

7. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define taille = 3`
- ☐ `#define taille = N`
- ☐ `#define N =  3`
- ☐ `#define N 3`

8. Un registre du processeur est :

- ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
- ☐ un composant qui contient la liste des fichiers du système
- ☐ une unité de calcul spécialisée de l'ordinateur
- ☐ une gamme de fréquence de fonctionnement du processeur

9. Sous unix (ou linux), la commande `ls` permet de :

- ☐ voir des clips musicaux
- ☐ afficher la liste de fichiers contenus dans un répertoire
- ☐ compiler un programme
- ☐ afficher le contenu d'un fichier texte

10. Quel est l'opérateur de différence en C :

- ☐ $\neq$
- ☐ `!`
- ☐ `!=`
- ☐ `<>`

11. Si cette erreur apparaît à la compilation :
`error: expected ';' before '}' token` que doit-on chercher dans le programme ?

- ☐ un point-virgule en trop
- ☐ une accolade manquante
- ☐ un point-virgule manquant
- ☐ une accolade en trop

12. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 5`

13. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 15
- ☐ 10
- ☐ 6
- ☐ 0

14. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int %d;`
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int k;`
15. Un bit est :
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme
    - ☐ un battement d'horloge processeur
    - ☐ la longueur d'un mot mémoire
16. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
17. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
printf("Majeur\n");
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Majeur`
- ☐ rien
- ☐ `Mineur`

18. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers` « `prinft` » que doit-on chercher dans le programme ?
    - ☐ un caractère interdit en C
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une variable non déclarée
    - ☐ une directive préprocesseur `#include` manquante
19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
    - ☐ certaines données de la mémoire de travail
    - ☐ des processus
    - ☐ en temps d'accès
    - ☐ les fichiers du disque
20. Un fichier source est :
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 points par réponse juste (unique) ; −0,5 points par réponse fausse. Durée : 20 minutes.

1. Un programme en langage C doit comporter une et une seule définition de la fonction :

   ☐ init

   ☐ include

   ☐ begin

   ☐ main

2. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

   ☐ `#appart <stdlib.h>`

   ☐ `#include <studio.h>`

   ☐ `#include <stdio.h>`

   ☐ `#include <studlib.h>`

3. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :

   ☐ 16

   ☐ 4

   ☐ 0

   ☐ 8

4. Sur unix (ou linux), la commande `mkdir` permet de :

   ☐ créer un répertoire

   ☐ créer un fichier texte

   ☐ changer de répertoire courant

   ☐ ouvrir un fichier texte

5. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :

   ☐ rien

   ☐ `Majeur Mineur`

   ☐ `Mineur`

   ☐ `Majeur`

6. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :

   ☐ mettre les blocs en séquence les uns à la suite des autres

   ☐ retourner un bloc

   ☐ répéter un bloc tant qu'une condition est vérifée

   ☐ sélectionner entre deux blocs à l'aide d'une condition

7. Sous unix (ou linux), la commande `cd` permet de :

   ☐ récupérer un programme arrêté avec la commande `ab`

   ☐ jouer de la musique

   ☐ changer de répertoire courant

   ☐ ouvir un bureau partagé (common desktop)

   ☐ détruire un fichier

8. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

   qu'est ce qui sera affiché ?

   ☐ `0 1 2 0 1 2 3`

   ☐ `0 0 1 1 2 2 3`

   ☐ `0 1 2 0 1 2`

   ☐ `0 1 2 3 0 1 2`

9. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction

   ☐ `int loop n;`

   ☐ `loop i;`

   ☐ `int k;`

   ☐ `int %d;`

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

    qu'est ce qui sera affiché ?

    ☐ `j = %d`

    ☐ `j = 0`

    ☐ `j = 4`

    ☐ `j = 5`

11. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

- ☐ le terminal affiche `5`
- ☐ le terminal affiche "Faux"
- ☐ le terminal affiche `x = 2`
- ☐ le terminal affiche `x = 5`

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

- ☐ la variable x vaut 16
- ☐ le programme affiche `****`
- ☐ la variable x vaut $-\frac{1}{2}$
- ☐ le programme affiche x

13. Une *segmentation fault* est une erreur qui survient lorsque :
    - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal
    - ☐ la division du programme en zones homogènes échoue
    - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
    - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée

14. Le code suivant :

```
int age = 18;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :
- ☐ `Majeur Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Mineur`

15. Le bus système sert à :
    - ☐ Arriver à l'heure en cours
    - ☐ transporter les processus du tourniquet au processeur
    - ☐ Écrire des données sur le dique dur
    - ☐ Transférer des données et intructions entre processeur et mémoire

16. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`

17. L'ordonnancement par tourniquet permet :
    - ☐ de doubler la mémoire disponible
    - ☐ d'afficher des ronds colorés à l'écran
    - ☐ de ne pas perdre de temps avec la commutation de contexte
    - ☐ d'entretenir l'illusion que les processus tournent en parallèle

18. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = N`
    - ☐ `#define taille = 3`
    - ☐ `#define N =  3`
    - ☐ `#define N 3`

19. Un bit est :
    - ☐ la longueur d'un mot mémoire
    - ☐ un battement d'horloge processeur
    - ☐ un chiffre binaire (0 ou 1)
    - ☐ l'instruction qui met fin à un programme

20. Pour l'extrait de programme suivant :

```
int produit = 0;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :
- ☐ 16
- ☐ 0
- ☐ 4
- ☐ 8

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le langage C est un langage
   - ☐ lu, écrit, parlé
   - ☐ interprété
   - ☐ composé
   - ☐ compilé

2. Le code suivant :

```
int i;
for (i = 0; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3
   - ☐ 0 1 2 3 4

3. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$

4. Une *segmentation fault* est une erreur qui survient lorsque :
   - ☐ le programme source a été enregistré sur le disque dur en plusieurs morceaux et l'un d'entre eux ne peut pas être chargé par le compilateur
   - ☐ le programme tente d'accèder à une partie de la mémoire qui ne lui est pas réservée
   - ☐ la division du programme en zones homogènes échoue
   - ☐ le programme tente d'afficher des caractères sur une ligne qui va au delà de la largeur de la fenêtre du terminal

5. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
   - ☐ analyse harmonique
   - ☐ analyse syntaxique
   - ☐ analyse sémantique
   - ☐ analyse lexicale

6. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
   - ☐ `int[] new tableau(5);`
   - ☐ `int toto[taille=5];`
   - ☐ `int toto[5];`
   - ☐ `char tableau[5];`
   - ☐ `int tab[] = 5;`

7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

   affichera :
   - ☐ `Majeur Mineur`
   - ☐ `Majeur`
   - ☐ `Mineur`
   - ☐ rien

8. Le code suivant :

```
int i;
for (i = 1; i < 5; i = i + 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1

9. Pour l'extrait de programme suivant :

```
int somme = 0;
int serie[4] = {2, 4, 10, 4};
for (i = 0; i < 4; i = i + 1)
{
  somme = somme + serie[i];
}
printf("somme = %d",somme);
```

   La valeur de somme affichée est :
   - ☐ 16
   - ☐ 3
   - ☐ 6
   - ☐ 20

10. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n,x,y");`
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x y);`

11. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`
    - ☐ `#include <stdio.h>`

12. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

La valeur affichée est :

- ☐ 8
- ☐ 16
- ☐ 4
- ☐ 0

13. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 3
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"

14. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3
- ☐ 4 3 2 1

15. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2 3`
- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2`

16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.c -o prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc -Wall prog.exe -o prog.c`

17. Quels calculs peut-on programmer en programmation structurée ?

- ☐ en programmation structurée on peut programmer tous les calculs programmables en langage machine
- ☐ certains programmes sont de vrais plats de spaghetti
- ☐ il y a des calculs programmables en langage machine et qui ne sont pas programmables en programmation structurée
- ☐ il y a des calculs programmables en programmation structurée qui ne sont pas programmables en langage machine

18. Sur unix (ou linux), la commande `mkdir` permet de :

- ☐ créer un fichier texte
- ☐ créer un répertoire
- ☐ changer de répertoire courant
- ☐ ouvrir un fichier texte

19. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ en temps d'accès
- ☐ les fichiers du disque
- ☐ des processus
- ☐ certaines données de la mémoire de travail

20. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = 4`
- ☐ `j = %d`
- ☐ `j = 5`
- ☐ `j = 0`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Le code suivant :

```
int age = 20;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Majeur Mineur`
- ☐ `Mineur`
- ☐ `Majeur`
- ☐ rien

2. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.

- ☐ `#define N =  3`
- ☐ `#define taille = N`
- ☐ `#define N 3`
- ☐ `#define taille = 3`

3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 0 1 0 1`
- ☐ `0 1 2 0 1 2`
- ☐ `0 0 1 1 2 2`
- ☐ `1 2 3 1 2`

4. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

- ☐ la case mémoire 8 contiendra 16
- ☐ le terminal affiche 8
- ☐ le bus explose
- ☐ la case mémoire 8 contiendra 0

5. Le langage C est un langage

- ☐ compilé
- ☐ lu, écrit, parlé
- ☐ composé
- ☐ interprété

6. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :

- ☐ des processus
- ☐ en temps d'accès
- ☐ les fichiers du disque
- ☐ certaines données de la mémoire de travail

7. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ `Mineur`
- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`

8. Le code suivant :

```
int i;
for (i = 8; i > 0; i = i - 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 8 6 4 2
- ☐ 8 6 4 2 0
- ☐ 0 2 4 6 8
- ☐ 8 2

9. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ le programme affiche "Faux"
- ☐ la variable y vaut 5
- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable x vaut 3

10. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
```

```
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 5`
- ☐ `j = 0`

11. Pour déclarer un tableau d'entiers de taille 5, on peut utiliser l'instruction
    - ☐ `int toto[taille=5];`
    - ☐ `int toto[5];`
    - ☐ `int[] new tableau(5);`
    - ☐ `int tab[] = 5;`
    - ☐ `char tableau[5];`
12. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ afficher le contenu d'un fichier texte
    - ☐ compiler un programme
    - ☐ voir des clips musicaux
13. Le code suivant :

```
int i;
for (i = 0; i < 7; i = i + 2)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4 5 6
- ☐ 0 1 2 3 4 5 6 7
- ☐ 0 2 4 6 8
- ☐ 0 2 4 6

14. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 4 3 2 1
- ☐ 0 1 2 3 4
- ☐ 1 2 3 4
- ☐ 4 3 2 1 0

15. Si cette erreur apparaît à la compilation : `Undefined symbols :"_prinft"` ou `référence indéfinie vers « prinft »` que doit-on chercher dans le programme ?
    - ☐ une faute de frappe dans un appel de fonction
    - ☐ une directive préprocesseur `#include` manquante
    - ☐ un caractère interdit en C
    - ☐ une variable non déclarée
16. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
    - ☐ `gcc -Wall prog.c -o prog.exe`
    - ☐ `gcc -Wall prog.exe -o prog.c`
    - ☐ `gcc prog.c -o -Wall prog.exe`
    - ☐ `gcc prog.exe -Wall -o prog.c`
17. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ chacun son tour, après que le processus précédent a terminé
18. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 0; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
      ...
    }
}
printf("j = %d\n", j);
...
}
```

qu'est ce qui sera affiché ?

- ☐ `j = %d`
- ☐ `j = 0`
- ☐ `j = 5`
- ☐ `j = 4`

19. Un fichier source est :
    - ☐ un document de référence du système
    - ☐ un document illisible pour les humains
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document qui doit être protégé
20. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
    - ☐ `#include <studio.h>`
    - ☐ `#include <stdio.h>`
    - ☐ `#appart <stdlib.h>`
    - ☐ `#include <studlib.h>`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. L'ordonnancement par tourniquet permet :
   - ☐ de ne pas perdre de temps avec la commutation de contexte
   - ☐ de doubler la mémoire disponible
   - ☐ d'entretenir l'illusion que les processus tournent en parallèle
   - ☐ d'afficher des ronds colorés à l'écran
2. Un fichier source est :
   - ☐ un document de référence du système
   - ☐ un document qui doit être protégé
   - ☐ un fichier texte qui sera traduit en instructions processeur
   - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
   - ☐ un document illisible pour les humains
3. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

   qu'est ce qui sera affiché ?
   - ☐ `1 2 3 1 2`
   - ☐ `0 1 0 1 0 1`
   - ☐ `0 0 1 1 2 2`
   - ☐ `0 1 2 0 1 2`
4. Un programme en langage C doit comporter une et une seule définition de la fonction :
   - ☐ main
   - ☐ init
   - ☐ include
   - ☐ begin
5. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
   `Undefined symbols :"_prinft"` ou
   `référence indéfinie vers « prinft »`
   - ☐ l'analyse des entrées clavier
   - ☐ l'analyse harmonique
   - ☐ l'analyse sémantique
   - ☐ l'édition de liens
6. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `new TP4`
   - ☐ `mkdir TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
7. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
8. Sous unix (ou linux), la commande `ls` permet de :
   - ☐ compiler un programme
   - ☐ afficher la liste de fichiers contenus dans un répertoire
   - ☐ voir des clips musicaux
   - ☐ afficher le contenu d'un fichier texte
9. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche "Faux"
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `int loop n;`
    - ☐ `loop i;`
    - ☐ `int %d;`
    - ☐ `int k;`
11. Le code suivant :

```
int somme = 0;
int i;
for (i = 1; i < 4; i = i + 1)
{
  somme = somme + i;
}
printf("%d", somme);
```

    affichera :
    - ☐ 1
    - ☐ 6
    - ☐ 42
    - ☐ 0
12. Après exécution du programme :

```
1  lecture 8 r0
2  valeur 3 r1
3  mult r1 r0
4  valeur 1 r2
5  add r2 r0
6  ecriture r0 8
7  stop
8  5
```

    - ☐ le terminal affiche 8
    - ☐ le bus explose
    - ☐ la case mémoire 8 contiendra 0
    - ☐ la case mémoire 8 contiendra 16

13. Le code suivant :

```
int i;
for (i = 4; i > 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 0 1 2 3 4
- ☐ 4 3 2 1
- ☐ 4 3 2 1 0
- ☐ 0 1 2 3

14. Sur unix (ou linux), la commande `mkdir` permet de :
    - ☐ créer un fichier texte
    - ☐ ouvrir un fichier texte
    - ☐ changer de répertoire courant
    - ☐ créer un répertoire
15. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ un point-virgule en trop
    - ☐ une accolade manquante
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop
16. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ tour à tour, un petit peu à chaque fois
    - ☐ tous ensemble
    - ☐ en parallèle, chacun dans un registre
    - ☐ chacun son tour, après que le processus précédent a terminé
17. Les lignes

```
int i;
int x=0;
for(i=0,i<5,i=i+1)
{
  x=x+1;
}
```

- ☐ comportent une erreur qui ne sera pas détectée
- ☐ comportent une erreur qui sera détectée au cours de l'analyse syntaxique
- ☐ comportent une erreur qui sera détectée au cours de l'édition de lien
- ☐ ne comportent aucune erreur

18. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :
    - ☐ `printf("x=%d et y=%d\n",x,y);`
    - ☐ `printf("x=%x et y=%y\n");`
    - ☐ `printf("x=%d et y=%d\n",x y);`
    - ☐ `printf("x=%d et y=%d\n,x,y");`
19. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse sémantique
    - ☐ analyse syntaxique
    - ☐ analyse harmonique
    - ☐ analyse lexicale
20. Afin de représenter la taille d'un tableau, définir une constante symbolique N valant 3.
    - ☐ `#define taille = 3`
    - ☐ `#define N 3`
    - ☐ `#define taille = N`
    - ☐ `#define N =  3`

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée

2. Après exécution jusqu'à la ligne 14 du programme C :

```
10  int main() {
11      int x = 5;
12
13      printf(" x  = %d\n", 2);
14
15      ...
16  }
```

   - ☐ le terminal affiche `x = 5`
   - ☐ le terminal affiche `x = 2`
   - ☐ le terminal affiche `5`
   - ☐ le terminal affiche "Faux"

3. La virtualisation de la mémoire permet notamment de stocker des portions inactives de la mémoire de travail sur le disque dur. Mais on perd :
   - ☐ en temps d'accès
   - ☐ des processus
   - ☐ les fichiers du disque
   - ☐ certaines données de la mémoire de travail

4. Un bit est :
   - ☐ un battement d'horloge processeur
   - ☐ un chiffre binaire (0 ou 1)
   - ☐ la longueur d'un mot mémoire
   - ☐ l'instruction qui met fin à un programme

5. Sur unix (ou linux), la commande `mkdir` permet de :
   - ☐ créer un répertoire
   - ☐ créer un fichier texte
   - ☐ ouvrir un fichier texte
   - ☐ changer de répertoire courant

6. Pour l'extrait de programme suivant :

```
int i;
int j;
for(i=4;i>0;i=i-1)
{
  for(j=i;j<6;j=j+1)
  {
    printf("*");
  }
  printf(" ");
}
```

   qu'est ce qui sera affiché ?
   - ☐ `**** **** **** ****`
   - ☐ `** *** **** *****`
   - ☐ `***** **** *** ***`
   - ☐ `** ** ** ** ** **`

7. Pour l'extrait de programme suivant :

```
int produit = 1;
int serie[4] = {2, 2, 2, 2};
for (i = 0; i < 4; i = i + 1)
{
  produit = produit * serie[i];
}
printf("produit = %d", produit);
```

   La valeur affichée est :
   - ☐ 16
   - ☐ 8
   - ☐ 0
   - ☐ 4

8. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

   affichera :
   - ☐ 4 3 2 1
   - ☐ 1 2 3 4
   - ☐ 4 3 2 1 0
   - ☐ 0 1 2 3 4

9. Pour compiler un programme `prog.c`, on utilise la ligne de commande :
   - ☐ `gcc -Wall prog.exe -o prog.c`
   - ☐ `gcc prog.exe -Wall -o prog.c`
   - ☐ `gcc prog.c -o -Wall prog.exe`
   - ☐ `gcc -Wall prog.c -o prog.exe`

10. Pour déclarer une variable qui sera utilisée comme variable de boucle on peut utiliser l'instruction
    - ☐ `loop i;`
    - ☐ `int loop n;`
    - ☐ `int %d;`
    - ☐ `int k;`

11. Si cette erreur apparaît à la compilation :
    `error: expected ';' before '}' token` que doit-on chercher dans le programme ?
    - ☐ une accolade manquante
    - ☐ un point-virgule en trop
    - ☐ un point-virgule manquant
    - ☐ une accolade en trop

12. Sur un ordinateur avec un seul processeur, habituellement les processus sont exécutés :
    - ☐ en parallèle, chacun dans un registre
    - ☐ chacun son tour, après que le processus précédent a terminé
    - ☐ tous ensemble
    - ☐ tour à tour, un petit peu à chaque fois

13. Un programme en langage C doit comporter une et une seule définition de la fonction :
    - ☐ include
    - ☐ init
    - ☐ main
    - ☐ begin
14. Un fichier source est :
    - ☐ un fichier texte qui sera traduit en instructions processeur
    - ☐ un document illisible pour les humains
    - ☐ un document de référence du système
    - ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur
    - ☐ un document qui doit être protégé
15. Quelle étape de la compilation vient d'échouer lorsqu'on a un message comme celui-ci :
    `Undefined symbols :"_prinft"` ou
    `référence indéfinie vers ≪ prinft ≫`
    - ☐ l'analyse harmonique
    - ☐ l'édition de liens
    - ☐ l'analyse sémantique
    - ☐ l'analyse des entrées clavier
16. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

    - ☐ le programme affiche `****`
    - ☐ la variable x vaut $-\frac{1}{2}$
    - ☐ le programme affiche x
    - ☐ la variable x vaut 16
17. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :
    - ☐ analyse syntaxique
    - ☐ analyse sémantique
    - ☐ analyse harmonique
    - ☐ analyse lexicale
18. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

    qu'est ce qui sera affiché ?
    - ☐ `0 1 2 0 1 2`
    - ☐ `0 1 2 0 1 2 3`
    - ☐ `0 0 1 1 2 2 3`
    - ☐ `0 1 2 3 0 1 2`
19. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
        ...
    }
}
printf("j = %d\n", j);
```

    qu'est ce qui sera affiché par ce printf ?
    - ☐ `j = %d`
    - ☐ `j = 5`
    - ☐ `j = 0`
    - ☐ `j = 4`
20. Un registre du processeur est :
    - ☐ une unité de calcul spécialisée de l'ordinateur
    - ☐ un composant qui contient la liste des fichiers du système
    - ☐ une case mémoire interne au processeur qui sera manipulée directement lors des calculs
    - ☐ une gamme de fréquence de fonctionnement du processeur

Licence 1 **Éléments d'informatique – contrôle continue** Prénom : Nom :
N° INE : .

Barème : 1 **points par réponse juste (unique) ;** $-0,5$ **points par réponse fausse. Durée :** 20 **minutes.**

1. Sous unix (ou linux), pour créer un répertoire TP4 dans le répertoire courant on peut utiliser la commande :
   - ☐ `mkdir TP4`
   - ☐ `new TP4`
   - ☐ `kwrite TP4`
   - ☐ `yppasswd`
2. Après exécution jusqu'à la ligne 15 du programme C :

```
10  ...
11  int main() {
12      int x = 5;
13
14      x = 3 * x + 1;
15
16      ...
17  }
```

   - ☐ la variable x vaut 16
   - ☐ le programme affiche `****`
   - ☐ le programme affiche x
   - ☐ la variable x vaut $-\frac{1}{2}$
3. Sous unix (ou linux), la commande `cd` permet de :
   - ☐ récupérer un programme arrêté avec la commande `ab`
   - ☐ détruire un fichier
   - ☐ ouvir un bureau partagé (common desktop)
   - ☐ changer de répertoire courant
   - ☐ jouer de la musique
4. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 2; i = i + 1)
{
    for (j = 0; j < 3; j = j + 1)
    {
        printf("%d ", j);
    }
}
```

qu'est ce qui sera affiché ?

- ☐ `0 0 1 1 2 2 3`
- ☐ `0 1 2 0 1 2`
- ☐ `0 1 2 3 0 1 2`
- ☐ `0 1 2 0 1 2 3`

5. Soit un programme contenant les lignes suivantes :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 5; j = j + 1)
    {
          ...
    }
}
printf("j = %d\n", j);
```

qu'est ce qui sera affiché par ce printf ?

- ☐ `j = %d`
- ☐ `j = 4`
- ☐ `j = 0`
- ☐ `j = 5`

6. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :
   - ☐ `#include <studlib.h>`
   - ☐ `#include <stdio.h>`
   - ☐ `#include <studio.h>`
   - ☐ `#appart <stdlib.h>`
7. Le code suivant :

```
int i;
for (i = 4; i >= 0; i = i - 1)
{
    printf("%d ", i);
}
printf("\n");
```

affichera :

- ☐ 1 2 3 4
- ☐ 4 3 2 1 0
- ☐ 4 3 2 1
- ☐ 0 1 2 3 4

8. Une de ces manière de composer les blocs de programmes ne fait pas partie des opérations de la programmation structurée :
   - ☐ mettre les blocs en séquence les uns à la suite des autres
   - ☐ répéter un bloc tant qu'une condition est vérifée
   - ☐ retourner un bloc
   - ☐ sélectionner entre deux blocs à l'aide d'une condition
9. Quel est l'opérateur de différence en C :
   - ☐ `!`
   - ☐ `!=`
   - ☐ $\neq$
   - ☐ `<>`
10. Sous unix (ou linux), la commande `ls` permet de :
    - ☐ afficher le contenu d'un fichier texte
    - ☐ afficher la liste de fichiers contenus dans un répertoire
    - ☐ voir des clips musicaux
    - ☐ compiler un programme
11. Pour l'extrait de programme suivant :

```
int i = 0;
int j = 0;
for (i = 0; i < 3; i = i + 1)
{
    for (j = 0; j < 2; j = j + 1)
    {
        printf("%d ", i);
    }
}
printf("\n");
```

qu'est ce qui sera affiché ?

- ☐ `0 1 2 0 1 2`
- ☐ `1 2 3 1 2`
- ☐ `0 1 0 1 0 1`
- ☐ `0 0 1 1 2 2`

12. Après exécution jusqu'à la ligne 15 du programme C :

```
10  int main() {
11      int x = 5;
12      int y = 3;
13
14      x = y;
15
16      ...
17  }
```

- ☐ la variable x vaut 5 et la variable y vaut 3
- ☐ la variable y vaut 5
- ☐ le programme affiche "Faux"
- ☐ la variable x vaut 3

13. Le code suivant :

```
int age = 15;
if (age < 18)
{
    printf("Mineur\n");
}
else
{
    printf("Majeur\n");
}
```

affichera :

- ☐ rien
- ☐ `Majeur`
- ☐ `Majeur Mineur`
- ☐ `Mineur`

14. Pour compiler un programme `prog.c`, on utilise la ligne de commande :

- ☐ `gcc -Wall prog.exe -o prog.c`
- ☐ `gcc prog.c -o -Wall prog.exe`
- ☐ `gcc prog.exe -Wall -o prog.c`
- ☐ `gcc -Wall prog.c -o prog.exe`

15. Soient deux variables entières `x` et `y` initialisées à 4 et 5 respectivement. L'affichage `x=4 et y=5` est obtenu avec la commande :

- ☐ `printf("x=%x et y=%y\n");`
- ☐ `printf("x=%d et y=%d\n,x,y");`
- ☐ `printf("x=%d et y=%d\n",x y);`
- ☐ `printf("x=%d et y=%d\n",x,y);`

16. Pour l'extrait de programme suivant :

```
int somme = 0;
for (i = 0; i < 5; i = i + 1)
{
  somme = somme + i;
  i = i + 1; /* attention ! */
}
printf("somme = %d",somme);
```

La valeur de somme affichée est :

- ☐ 6
- ☐ 15
- ☐ 10
- ☐ 0

17. Un bit est :

- ☐ l'instruction qui met fin à un programme
- ☐ un battement d'horloge processeur
- ☐ un chiffre binaire (0 ou 1)
- ☐ la longueur d'un mot mémoire

18. Laquelle des analyses suivantes ne fait pas partie des étapes de la compilation :

- ☐ analyse sémantique
- ☐ analyse lexicale
- ☐ analyse syntaxique
- ☐ analyse harmonique

19. Lorsqu'un programme utilise `printf` ou `scanf` il faut qu'il contienne l'instruction préprocesseur :

- ☐ `#include <studio.h>`
- ☐ `#include <studlib.h>`
- ☐ `#include <stdio.h>`
- ☐ `#appart <stdlib.h>`

20. Un fichier source est :

- ☐ un document illisible pour les humains
- ☐ un document qui doit être protégé
- ☐ un document de référence du système
- ☐ un fichier texte qui sera traduit en instructions processeur
- ☐ un fichier que l'ont doit citer dans les documents produits sur l'ordinateur