RHAYOUTE
Note: 7/20 (score total : 21.25/57)

+197/1/32+

QCM Langage C - Année 2020-2021
ENSEEIHT, 1SN
Katia Jaffrès-Runser.

Examen Session 1
21 janvier 2021

Durée : 60 minutes.

Les réponses sont attendues SUR LA DERNIERE PAGE DU SUJET qui est à rendre. Les réponses données sur les autres feuilles du sujet ne sont pas prises en compte lors de l'évaluation.

Les questions faisant apparaître le symbole ♣ peuvent présenter zéro, une ou plusieurs bonnes réponses. Les autres ont une unique bonne réponse.

Une question simple rapporte au maximum 1 point, une question multiple au maximum 3 points. Des points négatifs pourront être affectés à de *très mauvaises* réponses.

Pour valider un choix, il faut complètement **NOIRCIR** la case.

Seules les cases sont analysées, il vous est donc possible d'écrire ailleurs sans incidence sur votre rendu.

---

1 Structure d'un programme, constantes et types.

On considère le programme suivant qui calcule le périmètre d'un cercle connaissant son rayon :

```
1   int main(){
2     #include <stdlib.h>
3     #include <stdio.h>
4     float rayon = 3.4;  //rayon du cercle
5     char unite = 'c';
6     const float PI 3.1415;
7     printf("Le périmètre du cercle de rayon %f%c est %f%c\n",
8       rayon, unite, 2*PI*rayon, unite);
9     return EXIT_SUCCESS;
10  }
```

Question 1 ♣ Quelle(s) instruction(s) ne sont pas correctes :

- [ ] L'instruction de la ligne 1.
- [ ] L'instruction de la ligne 9.
- [x] L'instruction de la ligne 3.
- [x] L'instruction de la ligne 6.
- [ ] *Aucune de ces réponses n'est correcte.*

Question 2 Comment définir la constante préprocesseur MAX qui vaut 10 ?

- [ ] `#define MAX = 10;`
- [ ] `#define MAX 10;`
- [ ] `#DEFINE MAX = 10`
- [ ] `#const MAX 10`
- [ ] `#const MAX = 10;`
- [x] `#define MAX 10`

Question 3 Comment déclarer un nouveau type `t_tab` qui est un tableau de 30 booléens ?

- [ ] `define bool[30] t_tab;`
- [x] `typedef bool t_tab[30];`
- [ ] `define bool t_tab[30];`
- [ ] `typedef bool t_tab 30;`

Question 4 Comment définir `Str` comme un alias sur le type `char *` ?

- [x] `typedef char* Str;`
- [ ] `type Str is new char*;`
- [ ] `typedef Str char*;`
- [ ] `#define char* Str`

2 Variables et expressions

Question 5 ♣ Comment déclarer et initialiser une variable `x` de type `double` à 20 ?

- [ ] `double x += 20.0;`
- [x] `double x = 20.0;`
- [x] `double x = 20;`
- [ ] *Aucune de ces réponses n'est correcte.*

Question 6 Quelles valeurs donner à `XX1, XX2, XX3 et XX4` pour que le message « Bravo ! » s'affiche lors de l'exécution des instructions suivantes :

```
assert(XX1 == 5 - 2 * 5);
assert(XX2 == 25 % 10);
assert(XX3 == 25 / 10);
assert(XX4 == 25 / 10.0);
printf("%s", "Bravo !");
```

- [ ] A `XX1=-5, XX2=5, XX3=2.5 et XX4=2.5`
- [x] `XX1=-5, XX2=5, XX3=2 et XX4=2.5`
- [ ] B `XX1=15, XX2=5, XX3=2 et XX4=2`
- [ ] D `XX1=-5, XX2=2, XX3=5 et XX4=2.5`

Question 7 Quelle est la valeur de `a` après l'exécution des instructions suivantes :

```
int a = 10;
int b = 3;
a *= b;
```

- [ ] A 100
- [x] 30
- [ ] B 3
- [ ] E 10
- [ ] C Le programme ne compile pas
- [ ] F 13

Question 8 Comment déclarer une variable `n` de type entier ?

- [ ] A `n: int`
- [ ] C `n of int;`
- [x] `int n;`
- [ ] D `int n`

Question 9 ♣ Cet exercice s'intéresse au concept de masquage des variables. Soit le programme suivant (les inclusions de bibliothèque sont volontairement omises) :

```
1   int main() {
2     int alea = 20;
3     int diviseur = 2;
4     {
5         int alea = 3;
6         float diviseur = 2.0;
7         float res = alea / diviseur;
8         assert(res == 1.5);
9     }
10    int res = alea / diviseur;
11    assert(res == 10);
12    printf("%s", "Les tests passent");
13    return EXIT_SUCCESS;
14  }
```

Quelle(s) variable(s) sont masquée(s) dans cet exemple ?

- [x] `int diviseur` à la ligne 3 est masquée.
- [ ] C `float diviseur` à la ligne 6 est masquée.
- [x] `int res` à la ligne 10 est masquée.
- [ ] D *Aucune de ces réponses n'est correcte.*

3 Pointeurs

Question 10 ♣ Comment déclarer deux variables `a` et `b` de type pointeur sur caractère ?

- [X] `char *a, *b;`
- [X] `char* a; char *b;`
- [ ] C `char* a, b;`
- [X] `char * a, * b;`
- [ ] E *Aucune de ces réponses n'est correcte.*

Question 11 On suppose les instructions suivantes :

```
int var1 = 10;
int *p1 = &var1;
*p1 = 25;
printf("%d", var1);
```

Qu'affiche la dernière instruction ?

- [ ] A 10
- [ ] B l'adresse de `var1`
- [X] 25

4 Entrées/sorties

Question 12 ♣ La fonction `scanf` permet de lire des données typées entrées au clavier. Dans la suite, cocher les utilisations correctes de cette fonction. On supposera que les variables suivantes sont définies comme suit :

```
float prix; int val; char unite = 'm'; float peri; char nom[10];
```

- [●] `scanf("%s", &unite);`
- [X] `scanf("%f", &prix);`
- [ ] C `scanf("Le prix est %f", &prix);`
- [ ] D `scanf("%1.2d", peri);`
- [X] `scanf("%d %c", &val, &unite);`
- [●] `scanf("%s", &nom);`
- [X] `scanf("%s", nom);`
- [ ] H *Aucune de ces réponses n'est correcte.*

Question 13 La fonction `printf` permet d'écrire des données typées à l'écran. Dans la suite, cocher l'affichage que l'on observe à l'écran si les instructions suivantes sont exécutées (on suppose que la bibliothèque `stdio.h` est connue) :

```
float prix = 1.4; int quantite = 100; char devise = 'E';
printf("L'achat de %d %s revient à %1.3f%c.",
                        quantite, "ananas", quantite * prix, devise);
```

- [ ] A L'achat de 100 ananas revient à 140E.
- [ ] B L'achat de 100 ananas revient à 100 * 1.4E.
- [●] L'achat de 100 "ananas" revient à 140.000E.
- [X] L'achat de 100 ananas revient à 140.000E.

5 Structures de contrôle

Question 14 On suppose que `sequence1`, `sequence2` et `sequence3` représentent plusieurs instructions où chaque instruction se termine par un point-virgule. On suppose aussi que `cond1` et `cond2` sont des expressions booléennes. La conditionnelle :

```
Si cond1 Alors sequence1 SinonSi cond2 Alors sequence2 Sinon sequence3
```

peut alors s'écrire en C :

- [A] `if (cond1) then { sequence1 } else if (cond2) then { sequence2 } else { sequence3 }`
- [X] `if (cond1) { sequence1 } else if (cond2) { sequence2 } else { sequence3 }`
- [C] `if (cond1) { sequence1 } elif (cond2) { sequence2 } else { sequence3 }`

Question 15 ♣
Voici une fonction qui illustre la conditionnelle `Selon` :

```
int f(int n) {
    int r = 0;
    switch (n) {
    case 1:
       r += 1;
       break;
    case 2:
    case 3:
       r += 8;
       break;
    case 4:
    case 7:
       r += 10;
    case 10:
    case 13:
       r += 100;
    default:
       r -= 1;
    }
    return r;
}
```

Elle retourne une valeur de `r` qui dépend du paramètre `n`. Quels tests sont corrects parmi les propositions suivantes :

- [A] `assert(1 == f(0));`
- [B] `assert(99 == f(12));`
- [X] `assert(-1 == f(5));`
- [X] `assert(99 == f(13));`
- [X] `assert(1 == f(1));`
- [F] *Aucune de ces réponses n'est correcte.*

Question 16 ♣ Quelle(s) formulation(s) de la boucle `pour` sont juste(s) ? On supposera que `sequence` représente plusieurs instructions.

- [X] `for (int k = 100; k >= 0; k--){ sequence }`
- [B] `for (int j = 1; j +=2 ; j < 10 ){ sequence }`
- [C] `for (int i = 0, i < 10, i += 2){ sequence }`
- [X] `for (int j = 0; j <= 10 ; j += 2){ sequence }`
- [E] *Aucune de ces réponses n'est correcte.*

6 Enumération, Enregistrement et Tableau

Question 17 On souhaite définir un type énuméré qui représente les enseignes d'un jeu de cartes `pique, coeur, carreau, trèfle`. Quelle proposition retiendriez-vous pour cela ?

- [x] `enum Enseigne {PIQ, COE, CAR, TRE};`
- [ ] B `enum Enseigne is new {PIQ, COE, CAR, TRE};`
- [ ] C `Enseigne is enum {PIQ, COE, CAR, TRE};`

Question 18 Quelle proposition est vraie pour un type `enum Etat` qui représente les quatre états de la matières `LIQ, SOL, GAZ, PLA` dans cet ordre :

- [ ] A On a la relation d'ordre suivante : `LIQ > SOL > GAZ > PLA`.
- [ ] B `LIQ, SOL, GAZ, PLA` sont des chaines de caractère.
- [x] L'instruction `enum Etat etat1 = 1;` compile.

Question 19 On souhaite définir un type enregistrement qui représente une carte à jouer caractérisée par une enseigne de type `enum Enseigne` et une valeur entière. Quelle proposition retiendriez-vous pour cela ?

- [ ] A `Carte is new struct (enum Enseigne ens; int val;);`
- [x] `struct Carte { enum Enseigne ens; int val; };`
- [ ] C `Carte is struct (enum Enseigne ens; int val;);`

Question 20 ♣ Le pointeur `ptr_carte` est initialisé de la façon suivante :

```
struct Carte carte1 = {PIQ, 9};
struct Carte * ptr_carte = &carte1;
```

On souhaite afficher la valeur de `carte1` via le pointeur `ptr_carte`. Quelle(s) instruction(s) permet(tent) de le faire ?

- [x] `printf("%d", ptr_carte.val);`
- [x] `printf("%d", ptr_carte -> val);`
- [ ] C `printf("%d", ptr_carte.All.val);`
- [ ] D *Aucune de ces réponses n'est correcte.*

Question 21 On souhaite définir **un type** tableau qui représente un jeu de 52 cartes à jouer. Quelle proposition retiendriez-vous pour cela ?

- [ ] A `struct Carte Jeu[52];`
- [ ] B `typedef struct Carte[52] Jeu;`
- [x] `typedef struct Carte Jeu[52];`

Question 22 ♣ On souhaite initialiser la première carte d'un jeu avec l'as de pique. On suppose que la variable `jeu1` est déclarée comme suit `Jeu jeu1;`. Quelle(s) proposition(s) sont possibles ?

- [ ] A `jeu1(1).ens = PIQ; jeu1(1).val = 1;`
- [x] `jeu1[0].ens = PIQ; jeu1[0].val = 1;`
- [ ] C `jeu1[1].ens = PIQ; jeu1[1].val = 1;`
- [ ] D *Aucune de ces réponses n'est correcte.*

7 Sous-programmes

Question 23 On considère la portion de code suivante :

```
void p1(char *a) {
   ...
}
void p2(char *b) {
   char c;
   XXX
}
```

Cocher la valeur de `XXX` qui correspond à un appel possible de `p1`.

- [ ] `p1(c);`
- [ ] `p1(&b);`
- [x] `p1(&c);`

Question 24 Quelle est la signature qui correspond à une procédure `p` qui prend comme premier paramètre un entier `n` en mode `In` et comme deuxième paramètre un entier `d` en In/Out.

- [ ] `void p(int* n, int *d);`
- [x] `void p(int n, int *d);`
- [ ] `int p(int n, int d);`

Question 25 Soit le programme suivant :

```
#include <stdio.h>
int f1(int* valeur) {
   *valeur = *valeur / 10;
   return *valeur;
}
int main(){
   int donnee = 20;
   int donnee_retournee = f1(&donnee);
   printf("donnee :  %i ", donnee);
   printf("donnee_retournee :  %i ", donnee_retournee);
}
```

Quelles sont les valeurs de `donnee` et `donnee_retournee` à la fin du programme principal ?

- [x] `donnee = 2` et `donnee_retournee = 2`
- [ ] `donnee = 20` et `donnee_retournee = 20`
- [ ] `donnee = 20` et `donnee_retournee = 2`

8 Allocation dynamique

Question 26 ♣ L'allocateur `realloc` permet de modifier la taille mémoire allouée dynamiquement à une adresse donnée. Voici sa signature :

```
void* realloc(void* ptr_mem, size_t taille)
```

Cocher la ou les propositions justes :

- [x] `realloc` retourne `NULL` ou l'adresse d'une zone mémoire de `taille` octets.
- [ ] `ptr_mem` contient l'adresse de la zone mémoire après réallocation (mode in out).
- [x] Si `ptr_mem` vaut `NULL`, `realloc` se comporte comme `malloc`.
- [ ] `taille` représente l'incrément de taille mémoire demandé.
- [ ] *Aucune de ces réponses n'est correcte.*

Question 27 ♣ Cocher la ou les instructions correctes qui permettent d'allouer de l'espace pour enregistrer un caractère avec `malloc`.

- [x] `char *ch = malloc( sizeof(char) );`
- [x] `char *ch = malloc( sizeof(*ch) );`
- [ ] C `char *ch = malloc( char );`
- [ ] D `char ch = malloc( sizeof(char) );`
- [ ] E *Aucune de ces réponses n'est correcte.*

Question 28 Voici la définition de la procédure `malloc` :

```
void* malloc(size_t taille);
```

A quoi sert cette procédure ?

- [ ] A A allouer une zone mémoire de `taille` bits
- [ ] B A allouer une zone mémoire de `size_t` octets
- [x] A allouer une zone mémoire de `taille` octets

Question 29 On veut pouvoir enregistrer 15 caractères de plus dans un tableau de `T` caractères, tableau alloué dynamiquement. Un étudiant propose cette instruction qui compile et s'exécute sans erreur :

```
char *str = realloc(str, (T+15)*sizeof(char));
```

Pourquoi cet étudiant se trompe-t-il ?

- [ ] A Il faut indiquer uniquement `15 * sizeof(char)` en second paramètre de l'appel à `realloc`.
- [x] En cas d'échec de la réallocation, `realloc` retourne `NULL` et on aura perdu l'adresse de la mémoire initiale dans `tab`.

Question 30 Que signifie le fait que l'allocateur `malloc` retourne l'adresse `NULL` ?

- [x] L'allocateur n'a pas réussi à allouer la mémoire demandée de façon contigüe.
- [ ] B L'allocateur vous indique qu'il faut allouer de la mémoire dans la pile.

Question 31 ♣ Pour le jeu d'instructions suivant :

```
enum chat {SIAMOIS, CALICO, PERSAN, TABBY};
enum chat *my_cat;
my_cat = calloc(1, sizeof(enum chat));
assert(*my_cat == XXX);
```

Cocher une valeur pour `XXX` qui valide l'assert.

- [ ] A `NULL`
- [x] `0`
- [x] `SIAMOIS`
- [ ] D *Aucune de ces réponses n'est correcte.*

Question 32 ♣ Soient les instructions suivantes :

```
char *initiale = malloc(sizeof(char));
*initiale = 'A';
free(initiale);
printf("L'initiale est %c", *initiale);
```

Qu'affiche la dernière instruction `printf` ?

- [X] Ce code est faux.
- [X] Probablement 'A'
- [ ] `'A'`
- [ ] *Aucune de ces réponses n'est correcte.*
- [ ] L'exécution échoue à cause d'une erreur de segmentation

Question 33 Soient les instructions suivantes :

```
int *valeur = malloc(sizeof(int));
*valeur = 10;
```

Cocher la réponse qui permet de libérer la mémoire :

- [ ] `free(valeur, sizeof(int));`
- [X] `free(valeur); valeur = NULL;`
- [ ] `valeur.free();`
- [ ] `valeur = NULL;`

Question 34 ♣ On souhaite allouer ***** dynamiquement ***** une variable tableau de 15 caractères. Cocher la ou les bonne(s) instruction(s) :

- [ ] `char *t = 15 * malloc(sizeof(char));`
- [X] `char *t = malloc(15 * sizeof(*t));`
- [X] `char *t = malloc(15 * sizeof(char));`
- [X] `char *t = calloc(15, sizeof(char));`
- [ ] *Aucune de ces réponses n'est correcte.*

9 Les modules

Question 35 Le corps du module `date.c` présente la fonction suivante :

```
int max(int a, int b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}
```

Un programme principal `visualiser.c` inclut à la fois le module `date` et le module `maths.h` qui définit aussi une fonction `max`. Que faire pour pouvoir générer l'exécutable ?

- [X] Il faut rajouter le mot-clé `extern` devant `int max(int a, int b)`
- [X] Il faut rajouter le mot-clé `static` devant `int max(int a, int b)`

Question 36 Quelles sont les commandes pré-processeur qui permettent d'éviter l'inclusion multiple du module `date` ?

```
#ifndef DATE__H
#include DATE__H
...
# endif
```

```
#ifndef DATE__H
#define DATE__H
...
# endif
```

Question 37 Est-ce que la commande suivante produit un exécutable ? On suppose qu'il n'y a pas d'erreur dans les programmes.

```
c99 -Wextra -pedantic -c liste.c
```

Oui, si un sous-programme `int main()` existe dans `liste.c`. Non

Question 38 Un programmeur souhaite utiliser son propre module `pile` dans son programme principal décrit dans le fichier `principal.c`. Quelle instruction doit-on ajouter au début de `principal.c` ?

A `#import "pile.h"`
`#include <pile.h>`
`#include "pile.h"`

Question 39 On souhaite définir un module `fraction` en C. Quels fichiers doit-on créer par convention ?

Pour l'interface `fraction.c` et `fraction.h` pour le corps
B Pour l'interface `fraction.h` et `fraction.cc` pour le corps
Pour l'interface `fraction.h` et `fraction.c` pour le corps

10 Make

Question 40 Les premières règles d'un fichier `Makefile` sont les suivantes :

```
all:  test_file exemple_file

test_file:  test_file.o file.o
  c99 test_file.o file.o -o test_file

exemple_file:  exemple_file.o file.o
  c99 exemple_file.o file.o -o exemple_file
```

Quel(s) fichier(s) génère la commande `make all` ? On supposera que `make` n'a jamais été lancé.

A `test_file, test_file.o, file.o`
`test_file, exemple_file, test_file.o, file.o, exemple_file.o`
`test_file, exemple_file`

Question 41 Voici une des règles explicites listées dans un makefile :

```
date.o:  date.c date.h
  c99 -Wextra -pedantic -c date.c
```

Quelle est la commande exécutée par cette règle ?

A `date.o:  date.c date.h`
`c99 -Wextra -pedantic -c date.c`

Question 42 ♣ Soit la règle suivante :

```
a:b c
  xxx
```

La commande `xxx` sera exécutée :

- [X] si `c` est plus récent que `a`
- [X] si `b` est plus récent que `a`
- [C] si `a` est plus récent que `b`
- [D] *Aucune de ces réponses n'est correcte.*

+197/13/20+

Feuille de réponses :

Nom et prénom : RHAYOUTE Abdelmalek

Consignes :

Les réponses aux questions sont à donner exclusivement sur cette feuille : les réponses données sur les feuilles précédentes ne seront pas prises en compte.

Seules les cases sont analysées, il vous est donc possible d'écrire ailleurs sans incidence sur votre rendu.

Question 1 : [A] [B] [C] [D] [E]
Question 2 : [A] [B] [C] [D] [E] [F]
Question 3 : [A] [B] [C] [D]
Question 4 : [A] [B] [C] [D]
Question 5 : [A] [B] [C] [D]
Question 6 : [A] [B] [C] [D]
Question 7 : [A] [B] [C] [D] [E] [F] → correcteur
Question 8 : [A] [B] [C] [D]
Question 9 : [A] [B] [C] [D]
Question 10 : [A] [B] [C] [D] [E]
Question 11 : [A] [B] [C]
Question 12 : [A] [B] [C] [D] [E] [F] [G] [H]
Question 13 : [A] [B] [C] [D]
Question 14 : [A] [B] [C]
Question 15 : [A] [B] [C] [D] [E] [F]
Question 16 : [A] [B] [C] [D] [E] → correcteur.
Question 17 : [A] [B] [C]
Question 18 : [A] [B] [C]
Question 19 : [A] [B] [C]
Question 20 : [A] [B] [C] [D]
Question 21 : [A] [B] [C]
Question 22 : [A] [B] [C] [D]
Question 23 : [A] [B] [C]
Question 24 : [A] [B] [C]
Question 25 : [A] [B] [C]
Question 26 : [A] [B] [C] [D] [E]
Question 27 : [A] [B] [C] [D] [E]
Question 28 : [A] [B] [C]
Question 29 : [A] [B]
Question 30 : [A] [B]

Question 31 : [A] [X] [X] [D]

Question 32 : [X] [B] [C] [X] [E]

Question 33 : [A] [B] [X] [D]

Question 34 : [A] [X] [X] [X] [E]

Question 35 : [X]

Question 36 : [X]

Question 37 : [X]

Question 38 : [A] [X]

Question 39 : [B] [X]

Question 40 : [A] [X]

Question 41 : [A] [X]

Question 42 : [X] [X] [C] [D]