E3FR 3R-IN1A
Initiation à la programmation C
2021-2022

ESIEE PARIS

Examen 3R-IN1A

8 Décembre 2021 - durée 2 heures

DOCUMENTS ET TÉLÉPHONES INTERDITS

Remarques :
— Veuillez lire attentivement les questions avant de répondre.
— Le barème donné est un barème indicatif qui pourra évoluer lors de la correction.
— Rendez une copie propre et faites des réponses claires (servez vous de vos brouillons).
— Si vous ne savez pas, n'inventez pas ! Un malus de 25% sur le barème sera appliqué en cas de mauvaise réponse à l'une des questions à choix multiples(QCM).

Le sujet est composé de 4 pages

QCM et questions de cours (5 points)

1. Quelle est la sortie de ce programme ?

```c
#include <stdio.h>
int main(){
        int a = 3;
        int b = 7;
        int res = a++ + b++;
        printf( "%d - %d - %d", a, b,
            res );
        return 0;
}
```

Réponse A → 4 - 8 - 8
Réponse B → 4 - 8 - 10
Réponse C → 4 - 8 - 12
Réponse D → 4 - 8 - 16

2. Laquelle de ces lignes crée un tableau de 10 `int` ?
Réponse A → `int* tableau[10];`
Réponse B → `int tableau{10};`
Réponse C → `int tableau[10];`

3. Quelle est l'autre façon d'initialiser mon tableau avec ces valeurs ?

```c
int tableau[4];
tableau[0] = 10;
tableau[1] = 23;
tableau[2] = 505;
tableau[3] = 8;
```

Réponse A → `int tableau[4] = 10, 23, 505, 8;`
Réponse B → `int tableau[4] = [10, 23, 505, 8];`
Réponse C → `int tableau[4] = (10, 23, 505, 8);`
Réponse D → `int tableau[4] = {10, 23, 505, 8};`

4. Lequel de ces prototypes de fonction ne permet pas de faire passer un tableau en paramètre ?
Réponse A → `void fonction(int tableau[], int taille);`
Réponse B → `void fonction(int tableau, int taille);`
Réponse C → `void fonction(int *tableau, int taille);`

5. On suppose que `nombre` se trouve à l'adresse 5000, et `pointeur` à l'adresse 2500. Quelle sera la valeur affichée après la ligne 5.

```c
#include <stdio.h>
int main (){
        int nombre = 8;
        int *pointeur = &nombre;
        printf("%d \n", *pointeur);
        return 0;
}
```

Réponse A → 5000
Réponse B → 2500
Réponse C → 8
Réponse D → Impossible à prédire

6. On définit les constantes et les variables suivantes. Lesquelles de ces déclarations sont justes :

```c
#define A 5
#define B 7
#define C 5.6
int d;
int f = 7 ;
```

Réponse A → `int T1[10][B];`
Réponse B → `int T1[A][B];`
Réponse C → `float T2[C][B];`
Réponse D → `int T5[1][d];`
Réponse E → `int T6[A][f];`

7. Identifier les 8 erreurs de ce programme. Proposer une correction pour chacune des erreurs

```c
#include <stdio>
int main() {
        char c[10] = 'chaine';
        int m; n = 5;
        printf("c = %c\n", c);
        printf("Saisir un entier  ? ");
        scanf("%d", m);
        if (m = n)
                printf("m et n sont identiques \n");
        printf("somme = %d\n", m+n);
        return 0;
```

Exercices (15 points)

Exercice 1. (3 points)
On considère la suite numérique $u_{n,n\geqslant 1}$ définie par :

$$\begin{cases} u_1 &= 1 \\ u_2 &= 2 \\ u_n &= 2(u_{n-1} + u_{n-2}) \qquad \text{pour } n \geqslant 2 \end{cases}$$

1. Écrire une fonction *`suite()`* qui prend en paramètre un entier n et retourne l'entier u_n. Cette fonction devra être récursive.
2. Écrire une fonction *`affiche_suite()`* qui prend en paramètre un entier n et affiche les n premiers termes de la suite u_n, ainsi que leur somme.

Exercice 2. (5 points)
Soit `tab` un tableau d'entier de dimension M et qui contient n ($n << M$) éléments triés par ordre croissant.

```c
#include <stdio.h>
#define M 20

... insere_x(...);
... inverse_ordre(...);
... affiche_tab(...);

int main()
{

        int i,n = 10, x = 25;
        int tab[M] = {7, 17, 21, 23, 24, 26, 27, 30, 30, 38};
        affiche_tab(tab,n);
        if (M>n){
                i = insere_x(tab,n,x);
                affiche_tab(tab,n+1);
                printf("On insere l'entier %d  a l'indice %d \n",x,i);
                inverse_ordre(tab,n+1);
                affiche_tab(tab,n+1);
        }
        return 0;
}

... insere_x(...){...};
... inverse_ordre(...){...};
... affiche_tab(...){...};
```

Pour chacune des procédures et fonctions à écrire, il faudra utiliser un prototypage compatible avec le code ci-dessus.

1. Écrire une fonction *`insere_x()`* qui insère un entier x dans le tableau `tab` et retourne la position de l'entier ajouté. Par exemple, avec $x = 25$, on doit obtenir [7, 17, 21, 23, 24, 25, 26, 27, 30, 30, 38].
2. Écrire une procédure *`inverse_ordre()`* qui inverse l'ordre des éléments du tableau `tab`, c'est à dire en les classant par ordre décroissant.
3. Écrire une procédure *`affiche_tab()`* qui prend en paramètre un tableau d'entier et affiche ses éléments.

Exercice 3. (7 points)
Soit une classe de 20 élèves. Pour chaque élève on souhaites avoir les informations suivantes :

Nom	(chaîne de caractères)
Prénom	(chaîne de caractères)
Notes	(un tableau de 10 nombres réels)
Moyenne	(un nombre réel)

```c
#include <stdio.h>
#define NB_ELEVE_MAX 35
#define NB_MATIERE 10
#define NB_CARA_MAX 30

... Eleve_t ....

... saisie_noms (...);
... saisie_note (...);
... affiche_classe(...);
... calcul_moyenne_classe(...);
```

```
... affiche_moyennes(...);

int main ()
{
        int nb_eleve = 20;
        classe[NB_ELEVE_MAX];
        float moyenne_matiere[NB_MATIERE];
        float moyenne_generale;
        int i;
        for (i = 0; i<nb_eleve;i++){
                saisie_noms(&classe[i]);
                saisie_note(&classe[i]);
        }
        affiche_classe(classe,nb_eleve);
        moyenne_generale = calcul_moyenne_classe(classe, moyenne_matiere,
            nb_eleve);
        printf("La moyenne de la classe est %f \n",moyenne_generale);
        affiche_moyennes(moyenne_matiere);
        return 0;
}
... saisie_noms (...){...};
... saisie_note (...){...};
... affiche_classe(...){...};
... calcul_moyenne_classe(...){...};
... affiche_moyennes(...){...};
```

Attention, pour chacune des procédures et fonctions à écrire, il faudra utiliser un prototypage compatible avec le code ci-dessus.

1. Définir un nouveau type *Eleve_t* qui permet de stocker les données de chaque élève. Pour les tableaux et chaînes de caractères, on utilisera des tableaux de taille fixe.
2. Écrire une procédure *saisie_noms()* qui permet de saisir ou modifier les nom et prénom d'un élève. Cette fonction nécessite un passage de paramètre par adresse.
3. Écrire une procédure *saisie_note()* qui permet de saisir ou modifier les notes d'un élève et calculer sa moyenne générale. Attention les notes saisies devront être comprise entre 0 et 20. Cette fonction nécessite un passage de paramètre par adresse.
4. Écrire une procédure *affiche_classe()* qui permet d'afficher la liste des élèves de la classe (nom, prénom et moyenne générale)
5. Écrire une fonction *moyenne_classe()* qui retourne la moyenne générale de la classe et la moyenne pour chacune des matières. Attention, il y a plusieurs valeurs à "retourner", par conséquent un passage de variable par adresse sera nécessaire.
6. Écrire une procédure *affiche_moyennes()* qui affiche la moyenne de la classe pour chacune des matières.