

Examen de Programmation C (session 2)

Mardi 2 juillet 2019, de 10h30 à 12h30

Consignes

- Aucun document n'est autorisé.
- Les calculatrices, ordinateurs et téléphones portables sont interdits.
- Notez votre numéro d'étudiant en haut de chaque page.

Exercice 1. Deux points faciles

1 + 1 = 2 pts

- 1. Écrire un programme qui lit un entier au clavier, puis qui affiche l'entier inférieur et l'entier supérieur.
- 2. Modifier le programme pour indiquer si l'entier lu est supérieur, égal ou inférieur à 50.

Vous pouvez écrire un seul programme en indiquant quelles parties correspondent à la réponse 1 et 2.

Exercice 2. Deux boucles à un point

1 + 1 = 2 pts

- 1. Écrire un programme qui lit une valeur entière $n > 1$, puis qui affiche la suite de chiffres suivante, en utilisant que des boucles **for** :

$$[1][12][123][1234][12345] \cdots \underbrace{[123 \cdots n]}_n.$$

- 2. Modifier le programme pour utiliser des boucles **while**.

Exercice 3. Calcul des premières valeurs d'une suite

2 pts

Soit la suite u_n définie de la manière suivante pour $n > 0$:

$$u_i = \begin{cases} 0.25 & \text{si } i = 0 \\ 2 \cdot u_{i-1} \cdot (1 - u_{i-1}) & \text{sinon.} \end{cases}$$

- 1. Écrire un programme qui lit au clavier une valeur entière $n > 0$, puis qui calcule et affiche les valeurs u_i pour $i \in \{0, \dots, n\}$.

Exercice 4. Ré-inventons la racine carrée

0.5 + 1.5 = 2 pts

Soit $x > 0$ un nombre flottant. Le résultat de \sqrt{x} peut être calculé en utilisant la suite définie de la manière suivante :

$$r_{i+1} = 0.5 \cdot (r_i + x/r_i), \quad \text{avec } r_0 \text{ un nombre flottant donné.}$$

Plus particulièrement, si $r_0 \neq 0$, la suite r_i converge vers \sqrt{x} .

- 1. Déterminer r_3 pour $(x, r_0) = (3, 0.5)$.
- 2. Écrire un programme qui lit les valeurs flottantes (x, r_0) et un entier n au clavier puis, si r_0 est valide, calcule et affiche les valeurs r_i pour $i \in \{0, \dots, n\}$.

Exercice 5. Nombre d'occurrences dans un tableau

2 + 1 = 3 pts

Soit t un tableau d'entiers de taille 100.

- 1. Écrire un programme qui lit le contenu du tableau t , et la valeur d'un entier e , puis qui détermine et affiche le nombre d'éléments inférieurs à e dans le tableau t .
- 2. Écrire le prototype d'une fonction qui ferait le même traitement.

Exercice 6. À l'aide!!

1 + 1 = 2 pts

Un étudiant vous envoie le mail suivant : "Je ne comprends pas, mon programme C compile, mais il ne fait pas ce que je veux. Normalement, je devrais obtenir "a = 18 et b = 17". Mais je vous assure, il n'y a pas d'erreur!". Pour l'aider, vous lui demandez de vous envoyer son programme (ci-dessous).

```
void foo(int a, int b) {
    a = a - b;
    b = a - b;
    a = a + b;
}

int main(void) {
    int a = 17, b = 18;
    foo(a, b);
    printf("a= %d et b= %d \n", a, b);
    return 0;
}
```

- 1. Corriger le programme pour qu'il affiche effectivement le bon résultat, en expliquant chaque erreur.

Exercice 7. Les pointeurs

1 + 2 + 1 = 4 pts

- 1. Rappeler la définition d'un pointeur vue en cours.
- 2. Quel est le résultat de l'exécution du programme suivant. Donner une explication.

```
int
main(void) {
    int a = 17;
    int * ptr = &a;
    *ptr = a + 1;
    printf("--> a= %d et *ptr= %d \n", a, *ptr);
    a = (*ptr--) - 1;
    printf("--> a= %d et *ptr= %d \n", a, *ptr);
    return 0;
}
```

Soit le début de programme suivant.

```
int
main(void) {
    int tab[17];
    // ... partie a completer
    return 0;
}
```

- 3. Compléter le programme pour lire le contenu du tableau `tab` en utilisant l'arithmétique des pointeurs.

Exercice 8. Pour finir, on joue à la pétanque

1 + 2 + 0.5 + 1.5 = 5 pts

Ce sont les vacances, on joue à la pétanque! On considère que le cochonnet est en coordonnées (0,0), et que le tableau *T* contient les informations utiles sur chaque lancé de boule. Elles sont stockées sous forme d'une structure contenant les coordonnées (*x*, *y*) de la boule et le numéro (0 ou 1) de l'équipe. Le but est de connaître l'indice de la boule la plus proche du cochonnet, ainsi que l'équipe à laquelle elle appartient.

- 1. Écrire la définition de la structure `boule_t`.
- 2. Écrire une fonction qui prend un tableau *T* de taille 12 en paramètre, et qui retourne l'indice du lancé pour lequel la boule est la plus proche du cochonnet.
- 3. Écrire un programme principal qui permet d'utiliser cette fonction.
- 4. Que faut-il modifier (fonction et programme principal) pour pouvoir effectuer ce même traitement sur un table de taille *n*, avec *n* une valeur entière lue au clavier?