

## Examen de Programmation C

Vendredi 22 décembre 2017, de 14h à 16h

### Exercice 1. Structures conditionnelles

4 pts

La note de Programmation C se compose de la manière suivante : 25 % sur le contrôle continu, et 75 % sur l'examen final.

- 1. Écrire un programme qui lit votre note au contrôle continu et celle à l'examen final, puis qui calcule et affiche votre note finale.
- 2. Que faut-il ajouter au programme précédent pour indiquer si oui ou non vous aurez la moyenne à l'examen, c'est-à-dire, une note finale supérieure ou égale à 10?

L'appréciation que je mettrai sur votre copie sera la suivante :

- $18 \leq \text{note} \leq 20 \rightarrow$  "Excellent travail"
- $16 \leq \text{note} < 18 \rightarrow$  "Très bon travail"
- $13 \leq \text{note} < 16 \rightarrow$  "Bon travail"
- $10 \leq \text{note} < 13 \rightarrow$  "Travail correct"
- $8 \leq \text{note} < 10 \rightarrow$  "Vous y êtes presque"
- $0 \leq \text{note} < 8 \rightarrow$  "Travail insuffisant"

- 3. Modifier le programme pour afficher l'appréciation finale correspondant à votre note, en utilisant une structure de type **if-else**.
- 4. Reprendre la question précédente en utilisant une structure **switch-case**.

### Exercice 2. Deux boucles à un point

2 pt

- 1. Écrire un programme qui lit une valeur entière  $n$ , puis qui calcule et affiche  $i^2$ , pour  $i \in \{0, \dots, n\}$ , en utilisant une boucle **for**.
- 2. Modifier le programme pour utiliser une boucle **while**.

### Exercice 3. Retour sur la suite de Syracuse

2 pt

Soit  $n_0 > 0$  un nombre entier. Une suite de Syracuse est une suite définie de la manière suivante :

$$n_i = \begin{cases} n_{i-1}/2 & \text{si } n_{i-1} \text{ est pair,} \\ 3 \times n_{i-1} + 1 & \text{si } n_{i-1} \text{ est impair.} \end{cases}$$

- 1. Écrire un programme qui lit deux valeurs entières positives  $(n_0, k)$ , puis qui calcule la valeur  $n_k$  en utilisant la définition précédente.
- 2. Pour  $(n_0, k) = (15, 17)$ , quelle est la valeur de  $n_{17}$ ? Afficher les valeurs successives de la suite.

### Exercice 4. Boucle imbriquée et fonction

2 pt

- 1. Écrire une fonction qui prend une valeur entière en paramètre, puis qui affiche un motif comme celui ci-dessous pour  $n = 2$ .

```
XXXXXX XX
XXXXXX XX
XX XX XX
XX XX XX
XXXXXX XXXXXX
XXXXXX XXXXXX
```

### Exercice 5. Valeurs d'un tableau

2 pt

Soit  $t$  un tableau d'entiers de taille 20.

- 1. Écrire un programme qui lit le contenu de  $t$ , et une valeur entière  $e$ , puis qui détermine et affiche l'indice de la première occurrence de  $e$  dans  $t$ , si  $e$  appartient à  $t$ , et  $-1$  sinon.

### Exercice 6. Programme mystère

1 pt

- 1. Quel est le résultat de l'exécution du programme suivant?

```
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;
}
```

### Exercice 7. Tableau 1D et fonction

2 pt

- 1. Écrire une fonction qui prend en paramètre un tableau d'entiers  $t$  et sa taille  $n$ , et qui calcule et retourne la somme des éléments de  $t$ .
- 2. Écrire un programme principal qui permet d'utiliser cette fonction sur un tableau de taille 20 initialisé à la déclaration avec les valeurs 1, 2,  $\dots$ , 20.

### Exercice 8. Les pointeurs

3 pt

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

```
int main(void) {
    int a = 17;
    int * ptr = &a;
    *ptr = a + 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 9. Tableau 2D et fonction

4 pt

Soit  $t$  un tableau 2D de taille  $10 \times 10$ .

- 1. Écrire une fonction qui prend  $t$  en paramètre, qui détermine si chaque valeur de  $t$  est unique, c'est-à-dire, si pour tout  $(i, j)$ , il n'existe pas  $(i', j')$  tels que  $t[i][j] = t[i'][j']$ , et qui retourne un entier correspondant (0 pour faux et 1 pour vrai).
- 2. Écrire un programme principal qui permet d'utiliser cette fonction.
- 3. Que faut-il modifier (fonction et programme principal) pour pouvoir effectuer ce même traitement sur un tableau de taille  $n \times n$ , avec  $n$  une valeur entière lue au clavier.