Examen de langage C

L2 - info4A - UFR Sciences et Techniques - Université de Bourgogne

Session 1 - année 2023/2024

Modalités

Durée indicative : 1h30.

Documents autorisés (conjointement avec l'épreuve de C++) : 8 feuilles au format A4, recto-verso, avec contenu libre.

Calculatrices, ordinateurs et autres appareils communicants non autorisés.

Partie A

Soit la fonction suivante :

```
int f(int n)
{
    int i=0;
    for(; n >= 1; n = n/2)
    {
        i = i+1;
    }
    return i;
}
```

A1 (5%)

Réécrivez le code de cette fonction en utilisant une boucle `while` à la place de la boucle `for`. Le comportement de la nouvelle version doit être strictement identique.

A2 (5%)

Quelle est la valeur retournée par f(4) ?

Partie B

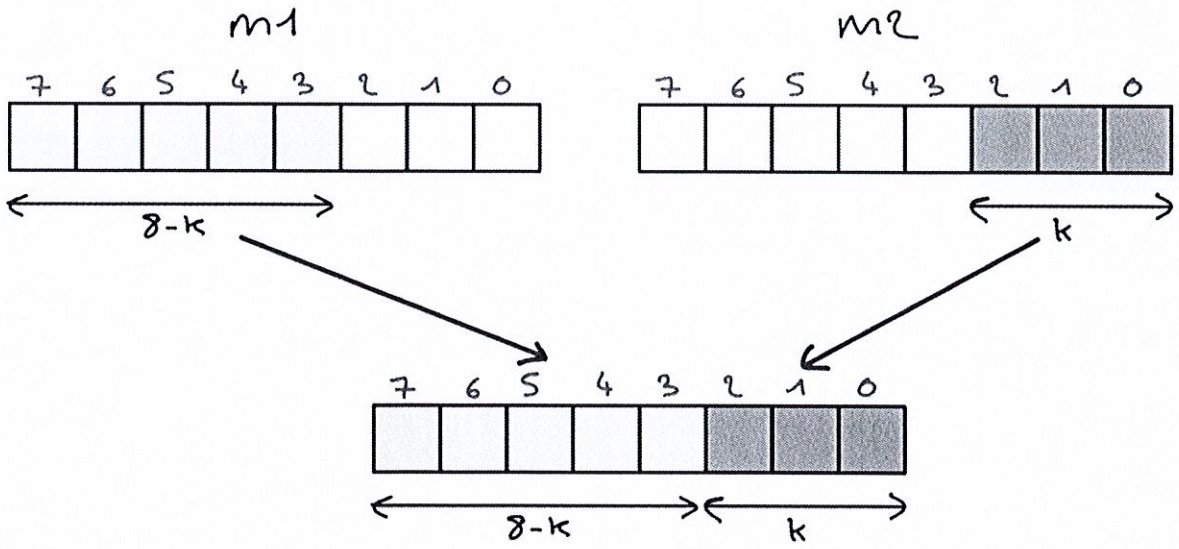
B1 (8%)

Donnez les valeurs affichées par l'exécution des lignes de code suivantes :

```
printf("%x ",0xE7 & 0x72);
printf("%x ",0xE7 | 0x72);
printf("%x ",0xE7 ^ 0x72);
printf("%x ",0xC0 >> 3);
```

B2 (12%)

Vous devez compléter la définition d'une fonction `combi` qui accepte en paramètre deux octets `m1` et `m2` et un entier `k` et qui retourne l'octet constitué des `8-k` bits les plus à gauche de `m1` et des `k` bits les plus à droite de `m2`.




```
uint8_t combi(uint8_t m1, uint8_t m2, int k)
{
    uint8_t masque1 = 0xFF << k;
    uint8_t masque2 = ~masque1;
    // À compléter
}
```

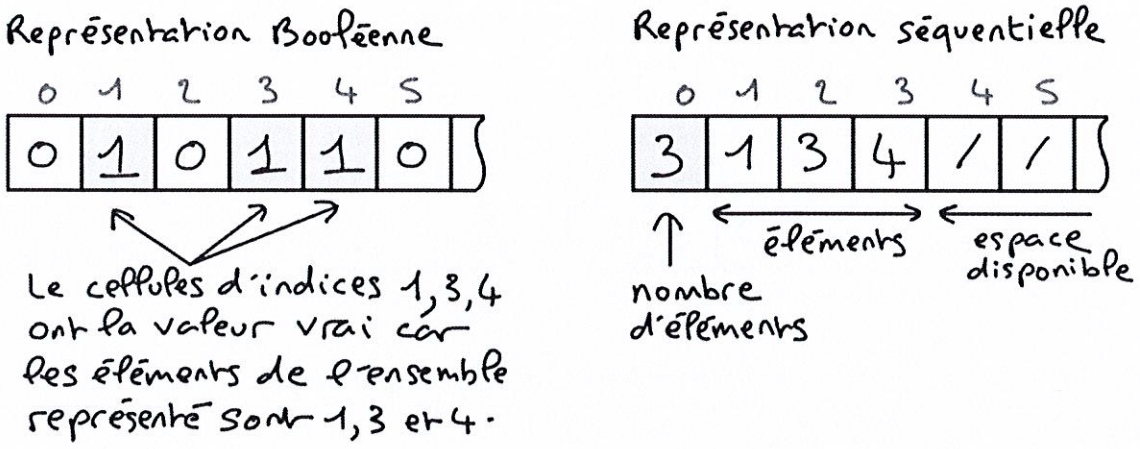
Sur la fiche de réponses, n'écrivez que les lignes manquantes.

Partie C

On considère deux représentations différentes d'ensembles d'entiers naturels compris entre 0 et 255. Dans la suite de l'énoncé, les lettres majuscules inclinées telles que E désignent des ensembles au sens mathématique, et non leur représentation informatique. La notation $|E|$ désigne le cardinal d'un ensemble E, c'est à dire son nombre d'éléments.

1. **Représentation Booléenne** : On représente un ensemble E (au sens mathématique) par un tableau `e` de 256 Booléens avec la convention suivante : pour tout entier $x \in 0..255$, $x \in E$ si et seulement si `e[x]` a la valeur *`vrai`*. Les Booléens sont eux-mêmes représentés par des entiers de type `uint8_t`. On utilise `1` pour *`vrai`* et `0` pour *`faux`*.
2. **Représentation séquentielle** : On représente un ensemble E par un tableau `e` de 257 entiers de type `uint8_t`. La première cellule contient le nombre d'éléments de E. Les $|E|$ cellules suivantes contiennent les éléments de E. Les cellules restantes ont des valeurs indéterminées.

Voici les deux représentations de l'ensemble $\{1, 3, 4\}$:



C1 (6%)

Donnez les lignes de code permettant de créer (dans la pile) et initialiser un tableau `t1` représentant l'ensemble $\{0, 2, 4\}$ sous forme Booléenne et un tableau `t2` représentant le même ensemble sous forme séquentielle.

C2 (14%)

Définissez la fonction `bool2seq` permettant de traduire une représentation Booléenne d'un ensemble en représentation séquentielle du même ensemble.

Le premier paramètre désigne la représentation Booléenne d'un ensemble. Le deuxième paramètre désigne un tableau existant, supposé être de taille suffisante, au contenu initial indéterminé, dans lequel la fonction doit mettre la représentation séquentielle du même ensemble.

```
void bool2seq(const uint8_t* input, uint8_t* output)
{
    // À compléter
}
```

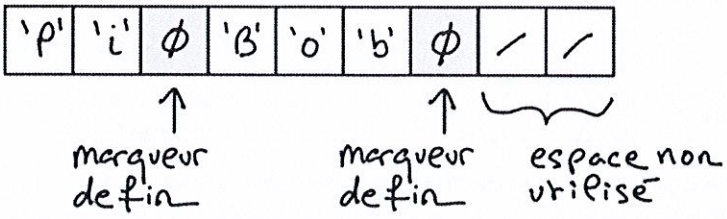
Merci de n'utiliser qu'une seule boucle permettant de parcourir la représentation Booléenne. Cette boucle permettra de rechercher les éléments de l'ensemble représenté **et** de les compter.

Partie D et E

DE1 (15%)

En langage C, une chaîne n'est pas un tableau de caractères. C'est une suite de caractères, terminée par un marqueur de fin, située **dans** un tableau de caractères. Il n'y a donc rien qui empêche de placer plusieurs chaînes à la suite dans un même tableau, dès lors que chaque chaîne est bien terminée par un marqueur de fin.

Dans cet exercice, nous allons placer dans un même tableau deux chaînes représentant respectivement le nom et le prénom d'une personne. La donnée constituée de ces deux chaînes sera appelée une **double chaîne**. À titre d'exemple, voici la représentation en mémoire de la double chaîne contenant le nom "Pi" et le prénom "Bob" dans un tableau de 9 cellules dont deux sont inutilisées.



Définissez la fonction `setdstr` permettant de placer dans un tableau de destination une double chaîne ayant pour nom et prénom deux chaînes passées en paramètre. Les rôles des paramètres sont les suivants :

- `nom` et `prenom` désignent les chaînes représentant les noms et prénoms à recopier dans la double chaîne à construire.
- `dest` désigne le tableau de destination, qui est supposé être de taille suffisante pour contenir la double chaîne à construire.

Vous ne devez **pas** utiliser de boucles, mais vous devez utiliser des fonctions standard telles que `strlen` et `strcpy`.

```
void setdstr(const char* nom, const char* prenom, char* dest)
{
    // À compléter
}
```

DE2 (15%)

Donnez les lignes de code permettant de créer **dans le tas** une double chaîne contenant le nom "Pi" et le prénom "Bob". (Ceci suppose de réserver un bloc mémoire de taille suffisante, puis placer la double chaîne dans ce bloc en utilisant la fonction `setdstr`.)

Partie F

F1 (10%)

Pour mémoire, on appelle **rationnel** tout nombre pouvant être représenté par une **fraction**, c'est à dire deux entiers appelés respectivement numérateur et dénominateur, comme par exemple $1/2$ ou $2/3$.

La structure suivante représente une fraction, et donc aussi un nombre rationnel :

```
typedef struct
{
    int num;
    int denum;
}ratio;
```

Donnez les lignes de code (qui pourraient par exemple se trouver dans la fonction `main`) permettant de réaliser les actions suivantes :

1. Créer une variable `r1` (située dans la pile) contenant une instance de `ratio` représentant la fraction $1/2$.
2. Créer **dans le tas** une instance non initialisée de `ratio`, pointée par une variable `r2` (elle-même située dans la pile).
3. Recopier dans l'instance de `ratio` pointée par `r2` le contenu de l'instance de `ratio` contenue dans `r1`.

F2 (10%)

Définissez la fonction `tabRatio` ...

```
ratio* tabRatio(int n, ratio* r)
{
    // À compléter
}
```

... qui crée dans le tas un tableau de `n` instances de `ratio` dont chaque cellule contient une copie de l'instance de `ratio` désignée par le paramètre `r`. La valeur de retour est un pointeur désignant le tableau créé.

Examen – Programmation C++

Exercice (5 points) :

Q1. Comment garantir que les membres d'une classe sont accessibles uniquement par les méthodes de cette classe ?
a) Déclarer les membres comme `protected`.
b) Déclarer les membres comme `private`.
c) Utiliser des fonctions amies.
d) Utiliser des méthodes virtuelles.

Q2. En C++, quel est l'effet de l'utilisation du mot-clé `const` à la fin de la déclaration d'une méthode de classe ?
a) La méthode ne peut modifier aucun membre de la classe.
b) La méthode peut être appelée sur des objets non-const.
c) La méthode doit être redéfinie dans chaque classe dérivée.
d) La méthode ne peut pas être appelée sur des objets const.

Q3. Quelle est l'erreur dans le code suivant de la surcharge de l'opérateur + ?

```
class Vecteur {
public:
    int x, y;
    Vecteur(int x, int y) : x(x), y(y) {}
    Vecteur operator+(const Vecteur& v) {
        x += v.x;
        y += v.y;
        return *this;
    }
};
```

a) L'opérateur devrait retourner un nouvel objet plutôt qu'une référence.
b) L'opérateur ne devrait pas modifier les membres de l'objet courant.
c) L'opérateur doit être défini en dehors de la classe.
d) L'opérateur devrait être une méthode statique.

Q4. Quel est le rôle de l'opérateur de résolution de portée : : en C++ ?
a) Il est utilisé pour accéder aux membres d'une classe depuis ses méthodes.
b) Il est utilisé pour définir des fonctions membres en dehors de la classe.
c) Il est utilisé pour accéder aux membres de l'espace de noms global.
d) Il est utilisé pour accéder aux membres d'une classe de base à partir de classes dérivées.

Q5. Quelle est l'erreur dans le code code suivant ?

```
class Carre: public Rectangle {
public:
    Carre(double c) : Rectangle(c, c) {}
};
```

a) La classe `Carre` ne devrait pas être dérivée de `Rectangle`.
b) Les arguments de la classe de base (`Rectangle`) doivent être mis entre crochets.
C) Le constructeur de `Carre` devrait appeler le constructeur de `Rectangle` avec deux arguments distincts.
d) Il n'y a aucune erreur dans le code.

Q6. La surcharge de fonction peut également être obtenue si deux fonctions ou plus ne diffèrent que par leur type de retour.
a) Vrai
b) Faux

Q7. Quand le destructeur est appelé ?
a) Quand un programme se termine
b) Quand une fonction se termine
c) Lorsque l'opérateur « `delete` » est utilisé
d) Toutes les réponses sont vraies

Q8. En C++, il est possible que la classe X, la classe Y et la classe Z héritent de la classe A.
a) Vrai
b) Faux