Examen de langage C

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

Session 2 - 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 l'algorithme suivant :

```
Fonction gen -> entier
  r <- 0
  Répéter 8 fois :
    r <- 2r
    Avec probabilité 1/2 faire : r <- r+1
  FinRépéter
  Retourner r
Fin
```

Cet algorithme initialise à 0 la variable `r` puis réalise réalise 8 itérations. À chaque itération, `r` est multipliée par 2, puis un tirage à pile ou face est simulé. Si la valeur pile est tirée (une chance sur deux), la valeur 1 est ajoutée à `r`. Dans tous les cas. À l'issue des 8 itérations, la valeur finale de `r` est retournée.

A1 (10%)

Implémentez la fonction `gen` en langage C.

Partie B

B1 (8%)

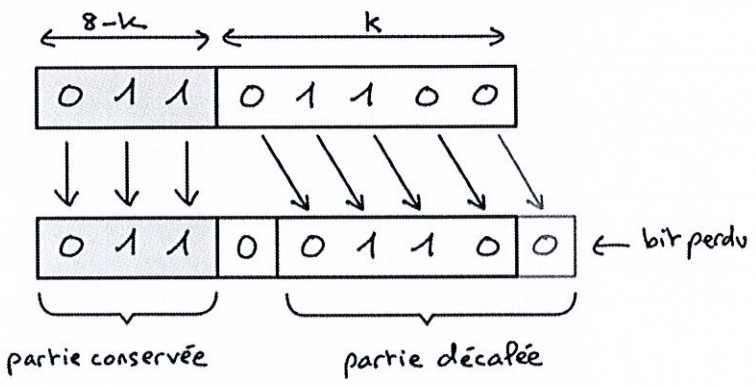
Donnez les valeurs affichées par l'exécution des lignes de code suivantes :

```
printf("%x ",0xFC & 0x3F);
printf("%x ",0xF0 | (3<<1));
printf("%x ",(~(0x0F)) & 0xFF);
printf("%x ",(0xFF >> 5) << 2);
```

B2 (12%)

Complétez la définition de la fonction `decFin` qui accepte en paramètres un octet `m` et un entier `k`, et qui retourne le mot binaire obtenu en décalant d'une position vers la droite les `k` derniers bits de `m`.

Voici un exemple avec `k` = 5 (Les 3 premiers bits sont conservés, les 5 suivants sont décalés à droite avec introduction d'un 0 à gauche).



Les deux masques initialement calculés sont importants pour réaliser une solution simple. Représentez-les au brouillon pour bien comprendre leur utilité et décomposez le problème en partant de cette base. N'utilisez pas de boucle.

```
uint8_t decFin(uint8_t m, int k)
{
    uint8_t masque1 = 0xFF << k;
    uint8_t masque2 = ~masque1;




}
```

Partie D

D1 (15%)

Donnez la définition de la fonction `cutp` qui accepte en paramètre une chaîne de caractères et tronque cette chaîne après le premier point (caractère `'.'`). La longueur du tableau contenant la chaîne ne doit pas être modifiée. Si la chaîne ne comporte pas de point, elle ne doit pas être modifiée.

Par exemple, les lignes de code suivantes...

```
char s[] = "Hello. You...";
cutp(s);
printf("%s\n",s);
```

...doivent produire l'affichage `Hello.`

```
void cutp(char* str)
{









}
```

Parties C et E

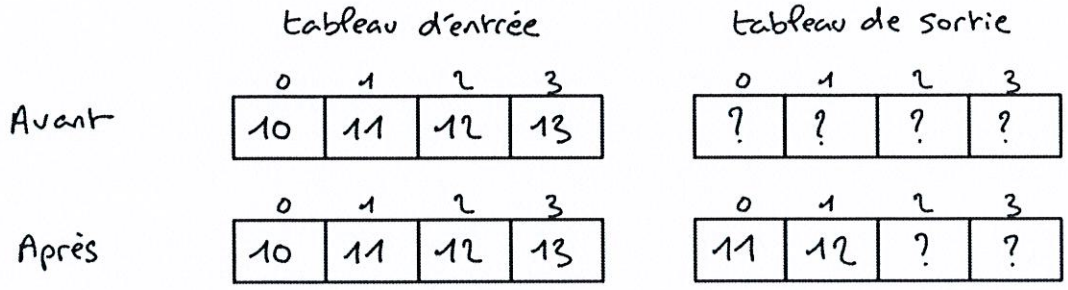
C1(10%)

Donnez la définition de la fonction...

```
void extract(const int* entree, int p1, int p2, int* sortie)
{









}
```

...qui place dans le tableau désigné par le paramètre `sortie` les valeurs situées inclusivement entre les positions `p1` et `p2` dans le tableau désigné par le paramètre `entree`. On suppose que le tableau de sortie existe et est de taille suffisante.

Par exemple, si le tableau d'entrée contient les valeurs 10, 11, 12, 13 et que p1 et p2 valent respectivement 1 et 2, alors après exécution de la fonction, les deux premières valeurs du tableau de sortie doivent être 11, 12. Les valeurs suivantes du tableau de sortie ne doivent pas être modifiées.



C2(10%)

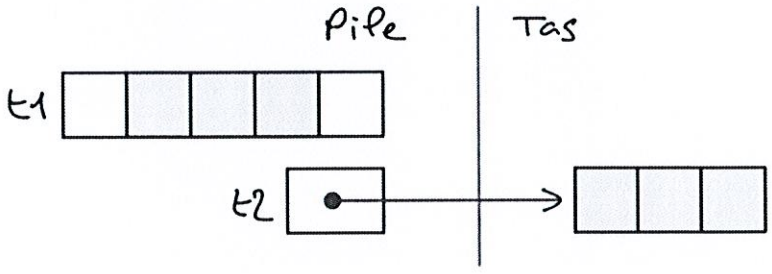
Donnez les lignes de code permettant de réaliser les actions suivantes :

1. Créer dans la pile un tableau de 6 entiers initialisé avec les valeurs 1,2,3,4,5,6.
2. Créer un autre tableau non initialisé de même taille.
3. Réaliser deux appels de la fonction `extract` de la question précédente pour placer dans le deuxième tableau les trois dernières valeurs du premier, suivies de ses trois premières valeurs.

À l'issue de l'exécution de ces ligne de code, le deuxième tableau devra donc contenir les valeurs 4,5,6,1,2,3.

E1 (10%)

Donnez les lignes de codes permettant d'obtenir la situation suivante en mémoire :



La tableau `t1` doit être rempli avec des entiers aléatoires compris inclusivement entre 0 et 9. Les trois éléments centraux de `t1` doivent être recopiés dans `t2` en utilisant la fonction `extract`.

E2 (5%)

Donnez les lignes de code permettant de libérer la mémoire réservée par l'exécution des lignes de codes de la question précédente.

Partie F

La structure `point` définie ci-dessous permet de représenter des points dans un espace discret à deux dimensions (les coordonnées de ces points sont des entiers).

```
typedef struct
{
    int x;
    int y;
}point;
```

La structure `lstP` définie ci-dessous permet de représenter des listes de points.

```
typedef struct
{
    int capa;     // Nombre maximum d'éléments
    int nbr;      // Nombre actuel d'éléments
    point* data;
}lstP;
```

La fonction suivante crée une liste de points dont la capacité est donnée en argument :

```
lstP* creeListe(int capa)
{
    lstP* w = (lstP*)malloc(sizeof(lstP));
    w->capa = capa;
    w->nbr = 0;
    w->data = (point*)malloc(capa * sizeof(point));
    return w;
}
```

F1 (6%)

Définissez la fonction...

```
int compPts(const point* p1, const point* p2)
{



}
```

...qui compare deux points et retourne 1 s'ils sont identiques (c'est à dire s'ils ont la même coordonnée `x` et la même coordonnée `y`), et 0 sinon.

F2 (6%)

Complétez la fonction...

```
int estDans(point p, const point* tab, int n)
{
    for(int i=0; i<n; i++)
    {
        if (                                    ) return 1;
    }
    return 0;
}
```

...qui retourne 1 si un point identique à `p` se trouve dans les `n` premières cellules du tableau désigné par `tab`, et 0 sinon. Vous devez utiliser la fonction `compPts` pour déterminer si deux points sont identiques.

F3 (8%)

Définissez la fonction...

```
void affListe(const lstP* w)
{




}
```

...qui affiche une liste de points. Chaque point doit être affiché sous la forme d'un couple tel que par exemple `(2,5)`.

L2 2023-2024

Examen Session 2 – Info4A

Numéro anonymat : _____

Exercice (5 points) :

1. Quelle est la différence entre une classe et un objet en C++ ?
☐ Une classe est une instance d'un objet
☐ Une classe est un pointeur vers un objet
☐ Un objet est une instance d'une classe

2. Dans la programmation orientée objet, l'encapsulation fait référence à :
☐ La possibilité d'accéder aux membres d'une classe depuis n'importe où dans le code
☐ Le fait de cacher les détails internes d'une classe et de fournir une interface publique pour interagir avec elle
☐ La possibilité d'hériter des fonctionnalités d'une autre classe

3. Qu'est-ce qu'un constructeur en C++ ?
☐ Une méthode qui permet d'accéder aux membres privés d'une classe
☐ Une méthode utilisée pour initialiser les membres d'une classe lors de la création d'un objet
☐ Une fonction membre statique d'une classe

4. Quelle est la signification du mot-clé "this" en C++ ?
☐ Il fait référence à l'objet actuel sur lequel une méthode est appelée
☐ Il est utilisé pour définir une constante en C++
☐ Il est utilisé pour déclarer une variable locale

5. Qu'est-ce que l'héritage multiple en C++ ?
☐ Le fait qu'une classe puisse hériter des membres de plusieurs classes de base.
☐ Le fait qu'une classe puisse être dérivée à la fois d'une classe de base et d'une classe dérivée
☐ Le fait qu'une classe puisse contenir plusieurs constructeurs.

6. Créer une classe Rectangle, avec des attributs pour sa largeur et sa hauteur, et des constructeurs sans et avec arguments.
Ajouter des méthodes pour calculer son périmètre et son aire. Le périmètre d'un rectangle est la somme de ses quatre côtés. L'aire d'un rectangle est le produit de sa largeur et de sa hauteur.

Rectangle.h

Rectangle.cpp