

**MPI, Info 111, Examen de deuxième session, 20 juin 2014 (trois heures)**

Documents autorisés : photocopie, TD, TP et résumé de la syntaxe C++ au dos de ce sujet. Calculatrices, et autres gadgets électroniques interdits. Le barème est indicatif.

Exercice 1 (Questionnaire à choix multiple, 5 points).

Répondre sur la dernière page du sujet. Les questions à choix multiple ont une unique bonne réponse. Des points négatifs seront affectés en cas de mauvaise réponse.

Qu'affichent chacun des fragments de programme suivants ?

Question 1

```
vector<int> t = {1, 9, -5, 3, 2};
int j = 0;
for (int i=0; i<t.size(); i++) {
    j = j + t[i];
}
cout << j << endl;
```

- A 3 B 9 C 0 D 10

Question 2

```
int j = 0;
for (int i=4; i>=2; i--) {
    j = j + i;
}
cout << j << endl;
```

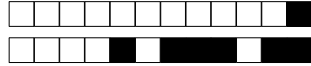
- A 9 B 10 C 0 D 3

Question 3

```
void f(int x, int y) {
    x = x+1;
    y = y-1;
}

int main() {
    int x,y;
    x = 1;
    y = 2;
    f(y,x);
    cout << x << " , " << y << endl;
}
```

- A 2,1 B 1,2 C 3,0 D *Erreur*

**Question 4**

```
int j = 0;
for (int i=0; i<4; i++) {
    j = j + i;
}
cout << j << endl;
```

- A 10 B 0 C 3 D 6

Question 5

```
int x, y;
x = 0;
y = 0;
while ( x < 10 ) {
    x = x + 1;
    y = 1;
}
cout << x << ", " << y << endl;
```

- A 9,9 B 0,1 C 10,1 D 9,1

Question 6

```
int j = 0;
for (int i=3; i<2; i++) {
    j = j + i;
}
cout << j << endl;
```

- A 3 B 9 C 10 D 0

Question 7

```
int syracuse(int n) {
    int resultat = 0;
    while (n != 1) {
        if (n % 2 == 0) // n est pair?
            n = n/2;
        else
            n = 3*n+1;
        resultat++;
    }
    return resultat;
}

int main() {
    cout << syracuse(1) << ", " << syracuse(3) << endl;
}
```

- A 1,7 B 1,8 C 0,7 D 0,6

**Question 8**

```
int x, y;
y = -3;
x = -y;
if ( y > 0 ) {
    y = 2*x;
} else {
    y = -y;
}
cout << x << ", " << y << endl;
```

- A 6, -3 B 6, 3 C -3, -3 D 3, 3

Exercice 2 (Boucles et conditionnelles : 3 points).

Rappel : Les mois de janvier, mars, mai, juillet, août, octobre et décembre ont 31 jours. Février a eu 28 jours en 2014. Tous les autres mois ont 30 jours.

1. Écrire un programme qui affiche tous les jours de l'année 2014 sous la forme suivante :

```
1/1/2014
2/1/2014
3/1/2014
...
31/12/2014
```

2. Modifier le programme précédent pour que l'affichage soit de la forme développée suivante :

```
Mercredi 1 janvier 2014
Jeudi 2 janvier 2014
Vendredi 3 janvier 2014
...
Mercredi 31 décembre 2014
```

Exercice 3 (Fonctions).

Implanter la fonction dont le prototype, la documentation et les tests sont donnés ci-dessous :

```
/** Somme des cubes
 * @param n un entier positif
 * @return 1^3+2^3+...+n^3
 */
int sommeDesCubes(int n);

void sommeDesCubesTest() {
    ASSERT( sommeDesCubes(0) == 0 );
    ASSERT( sommeDesCubes(1) == 1 );
    ASSERT( sommeDesCubes(2) == 9 );
    ASSERT( sommeDesCubes(3) == 36 );
}
```

**Exercice 4** (Fonctions, tests, tableaux : 3 points).

On considère la fonction `mystere` suivante :

```
vector<int> mystere(vector<int> bar) {
    for (int i=0; i+1<bar.size()-1; i++) {
        bar[i] = bar[i+1];
    }
    bar[bar.size()-1] = 0;
    return bar;
}
```

1. Exécuter pas-à-pas cette fonction sur deux exemples de votre choix. Préciser l'état des variables à chaque étape, ainsi que la valeur renvoyée.
2. Deviner ce que cette fonction est sensée faire. Donner des noms informatifs à la fonction et à ses variables. Écrire la documentation et quelques tests.
3. Pour un tableau de taille n , combien d'affectations sont effectuées ?

Exercice 5 (Fonctions, complexité).

1. Implanter la fonction dont le prototype, la documentation et les tests sont donnés ci-dessous :

```
/** La fonction exponentielle
 * @param x un nombre flottant
 * @param k un nombre entier positif
 * @return une approximation de exp(f) calculée avec la formule
 *         1 + x + x^2/2 + ... + x^k/k!
 */
double exp(double x, int k);

void expTest() {
    double PRECISION = 0.00001;
    ASSERT( abs( exp(0, 10) - 1 ) <= PRECISION );
    ASSERT( abs( exp(1, 20) - 2.71828 ) <= PRECISION );
    ASSERT( abs( exp(2, 20) - 7.38905 ) <= PRECISION );
    ASSERT( abs( exp(1, 0) - 1 ) <= PRECISION );
    ASSERT( abs( exp(1, 1) - 2 ) <= PRECISION );
}
```

On rappelle que la fonction `double abs(double x)` renvoie la valeur absolue de x .

2. Déterminer la complexité de cette fonction. Pour cela, on se placera dans le modèle de calcul où k est la mesure de la taille du problème, et où on comptera les opérations élémentaires '+' et '*'.

**Exercice 6** (Tableaux à deux dimension).

On considère la fonction `mystere` suivante, ainsi que sa fonction de test :

```
int mystere(vector<vector<int>> bidule, int f, int a) {
    int zou = 0;
    for (int i=f-1; i <= f+1; i++)
        for (int j=a-1; j<= a+1; j++)
            zou = zou + bidule[i][j];
    return zou/9;
}

void testMystere() {
    vector<vector<int>> flop = { { 3, 3, 0 },
                               { 3, 3, 0 },
                               { 6, 0, 0 } };
    ASSERT( mystere(flop, 1, 1) == 2 );

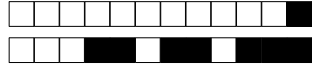
    ASSERT( mystere(flop, 0, 0) == 3 );
    ASSERT( mystere(flop, 0, 1) == 2 );
    ASSERT( mystere(flop, 1, 0) == 3 );
    ASSERT( mystere(flop, 1, 2) == 1 );
    ASSERT( mystere(flop, 2, 0) == 3 );
    ASSERT( mystere(flop, 2, 1) == 2 );
}
```

1. Est-ce que le premier test passe ?
2. Deviner ce que la fonction `mystere` est sensée faire. Donner des noms informatifs à cette fonction et ses variables. Écrire la documentation.
3. Que se passe-t-il lors de l'exécution des tests suivants ?
4. (optionnel) Proposer une modification de la fonction `mystere` pour que tous les tests passent.
5. Implanter la fonction suivante :

```
/** filtre de lissage
 * @param image une image en niveau de gris
 * @return une copie de l'image, lissée en utilisant
 *         la fonction de moyennage précédente
 */
vector<vector<int>> lisseImage(vector<vector<int>> image);
```

On pourra faire l'hypothèse que la fonction `mystere` se comporte correctement aux bords.

6. Déterminer la complexité de cette fonction, en précisant le modèle de calcul (taille du problème, opérations élémentaires).



+1/6/55+



Feuille de réponses, Questionnaire à Choix Multiples, Info 111, 20 juin 2014

À détacher et rendre séparément. Les réponses aux questions à choix multiple sont à donner exclusivement sur cette feuille : les réponses données sur les feuilles précédentes ne seront pas prises en compte.

Écrire votre nom et prénom ci-dessous, et coder votre numéro d'étudiant :

Nom et prénom :

- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9

- Question 1 : A B C D
- Question 2 : A B C D
- Question 3 : A B C D
- Question 4 : A B C D
- Question 5 : A B C D
- Question 6 : A B C D
- Question 7 : A B C D
- Question 8 : A B C D



Annexe : résumé de la syntaxe de base C++

Les exemples suivants résument la syntaxe des instructions de base C++, et précisent les conventions de codage utilisées dans le cadre de ce module : indentation, espacement, documentation au format javadoc¹ et tests.

```
#include <iostream>           // Squelette de programme
using namespace std;
int main () {
    ...
}
```

```
cin >> n;                    // Lit la variable n au clavier
cout << 3*x+1;                // Affiche la valeur d'une expression
cout << endl;                 // Affiche un saut de ligne
```

```
if ( x == 1 ) {              // Instruction conditionnelle
    ...;
} else if ( x < 2 and not y > 3 ) {
    ...;
} else {
    ...;
}
```

```
for ( int i=0; i < 10; i++ ) { // Instruction itérative : boucle for
    ...;
}
```

```
while ( i <= 10 ) {          // Instruction itérative : boucle while
    ...;
}
```

```
do {                          // Instruction itérative : boucle do ... while
    ...;
} while ( i <= 10 );
```

```
/** La fonction factorielle
 * @param n un nombre entier positif
 * @return n!
 */
int factorielle(int n) {
    int resultat = 1;
    for ( int k = 1; k <= n; k++ ) {
        resultat = resultat * k;
    }
    return resultat;
}

/** Les tests de la fonction factorielle */
void factorielleTest() {
    ASSERT( factorielle(0) == 1 );
    ASSERT( factorielle(1) == 1 );
    ASSERT( factorielle(2) == 2 );
    ASSERT( factorielle(3) == 6 );
    ASSERT( factorielle(4) == 24);
}
```

1. Pour plus de détails sur javadoc, voir par exemple <http://fr.openclassrooms.com/informatique/cours/presentation-de-la-javadoc/les-tags-javadoc-1>