IFT1166 - Session Hiver, Intra

Mohamed Lokbani

# IFT1166 - INTRA

Inscrivez tout de suite votre nom et code permanent.

Nom: _____| Prénom(s): _____|

Signature: _____| Code perm: _____|

Date : 8 mars 2005

Durée: 2 heures (de 18h30 à 20h30) Local: Z-330 du Pavillon Claire-McNicoll (ancienne aile Z).

## Directives:

- Toute documentation permise.
- Calculatrice **non** permise.

- Répondre directement sur le questionnaire.
- Les réponses **doivent être brèves, précises et clairement présentées.**

1._____/15 (1.1 a, b ; 1.2 a, b)

2._____/15 (2.1, 2.2, 2.3, 2.4, 2.5)

3._____/20 (3.1)

4._____/15 (4.1)

5._____/35 (5.1)

Total:_____/100

**Question 1 (15 points)** Soit la fonction suivante :

```
int& UneFonction(int &k) {
    return k;
}
```

**1.1** Soit la déclaration « `int i = 20 ;` ».

**a)** Quelle sera la valeur de la variable ***i*** après l'exécution du code suivant :

```
UneFonction(i)=42;
```

***i* =**
**Pourquoi?** (Courte explication)

**b)** L'instruction suivante est-elle correcte ? Si oui, que fait-elle au juste ?

```
UneFonction(i)++;
```

**Correct** ☐ **Incorrect** ☐

**Pourquoi?** (Courte explication)

**1.2** Supposer maintenant que le prototype de la fonction « UneFonction » est comme suit : **`int& UneFonction(int k)`** ; le code de la fonction reste le même. Sachant aussi que « `int g = 52;` »

**a)** Quelle sera la valeur de la variable ***j*** dans ce qui suit :

```
int j = UneFonction(g);
```

***j* =**
**Pourquoi?** (Courte explication)

**b)** L'instruction suivante est-elle correcte ? Si oui, que fait-elle au juste ?

```
int j = UneFonction(g)++;
```

**Correct** ☐ **Incorrect** ☐

***j* =**

**Pourquoi?** (Courte explication)

---

**Question 2 (15 points)** Soient les deux fonctions :

**Test -1-**

```
void Test(char *msg, int cnt = 3, char *post = "!"){
        cout << msg;
        while(cnt--) cout << post;
        cout << endl;
}
```

**Test -2-**

```
void Test(char *msg, char *post){
        Test(msg, 3, post);
}
```

Tout en justifiant votre réponse, quelle fonction « `Test` - ?-» est appelée dans les cas 4.1, 4.2 et 4.3. Et que va-t-elle afficher en sortie?

**2.1** `Test("Salut");`

**Test -1-** ☐

**Test -2-** ☐

**2.2** `Test("Huh","?");`

**Test -1-** ☐

**Test -2-** ☐

**2.3** `Test("Arrrggggg", 5, "h");`

**Test -1-** ☐

**Test -2-** ☐

**2.4** Est-ce que la fonction « Test-2- » est une fonction récursive?

**2.5** Que se passera-t-il pour l'appel « `Test("Salut");` » si on changeait le prototype de « Test -2- » comme suit : `void Test(char *msg, char *post = "!")`?

**Question 3 (20 points)** Identifier **toutes** les erreurs de **syntaxe** et de **logique** dans le code suivant.

```
1  include <iostream,h>
2
3  typedef true = 1;
4  typedef false = 0;
5  const int Boolean;
6
7  vold main(){
8     char quitter;
9     char montab{20};
10
11    while (quitter = false) {
12       cout << "Entrer q pour quitter ou c pour continuer" << endl;
13       cin >> quiltter;
14
15       if (quitter != q) init_arr(montab);
16       else compteur++;
17    }
18    cout << Compte " << compteur << " passes " endl;
19 }
20
21 int init_arr(char arr[20]);
22
23 int init_arr(char arr[20]);{
24    int arr = 7;
25    for (int ascii = 32; ascii < 52; ascii++) {
26        arr[i] = char(ascii);
27    }
28    return(arr);
29 }
```

Les réponses doivent être sous cette forme :

Ligne No : ……
Erreur : ……
Courte explication et suggestion de solution si c'est possible: …….

**Question 4 (15 points)** On suppose les déclarations suivantes :

```
const int MAXSIZE = 80;

typedef char StringTab[MAXSIZE];

typedef struct {
           StringTab chaine;
           int       taille_actuelle;
} StringType;

void StringEncrypte(StringType str1, StringType str2);
```

Vous devez écrire le code de la fonction **`StringEncrypte`**, qui crypte chacun des caractères du premier argument, **`str1`**, de la manière suivante :

Pour tout **`i`**, la nouvelle valeur de **`str1[i]`** sera égale à :

**`char ((int(str1[i]) + int(str2[i])) % 128)`**.

Si la chaîne **`str1`** est plus longue que la chaîne **`str2`**, le caractère espace blanc prendra la place des caractères manquants de **`str2`**.

**Question 5 (35 points)** Écrire un programme complet en C++ qui calcule les diviseurs positifs d'un nombre entier. Le résultat sera affiché sur une seule colonne et par ordre décroissant.

Au lancement, le programme doit décrire comment il doit être utilisé. Par la suite, il demande à l'usager d'entrée un nombre entier. Le programme doit permettre à l'utilisateur de répéter ce processus autant de fois qu'il désire. Le programme se comporte comme suit, si :

-1- L'utilisateur entre un nombre entier négatif ou nul. Dans ce cas, vous devez indiquer à l'utilisateur que les valeurs entrées sont erronés et que vous n'acceptez que des valeurs positives. Puis vous donnez la chance encore une fois à l'usager d'entrer une valeur correcte. Vous bouclez sur ce problème tant qu'il n'a pas inséré une valeur correcte.

-2- Après avoir entré une valeur positive non nulle, puis calculé les diviseurs de cette valeur ; le programme peut demander à l'usager s'il veut entrer une nouvelle valeur ou terminer le programme. La réponse doit être du style `O` ou `o` (pour oui) et `N` ou `n` (pour non). Si l'usager entre par exemple le caractère « g », vous devez l'informer que vous n'acceptez comme réponse que les lettres **O**, **o**, **N**, ou **n**.

L'algorithmique pour calculer les diviseurs d'un nombre est toute simple :

Soient **x** et **y** deux variables de type `int`. L'expression **x%y** retourne le reste de la division entière de **x** par **y**. Pour calculer les diviseurs de **x**, on essaie de diviser **x** par tous les entiers compris entre `1` et **x**. Si le reste de la division **x/y** est nul, alors l'entier **y** est un diviseur de **x**. Par exemple, les diviseurs de `60` sont : `{1 2 3 4 5 6 10 12 15 20 30 60}`. L'entier **25** n'est pas un diviseur de **60** car le reste de la division de **60/25** n'est pas nul.

**La simplicité de votre code et la clarté de sa présentation sont des éléments importants de la notation**