IFT1166 - Session Hiver, Intra

Mohamed Lokbani

IFT1166 - INTRA

Inscrivez tout de suite votre nom et code permanent.

Nom: _____| Prénom(s): _____|

Signature: _____| Code perm: _____|

Date : 28 février 2004

Durée: 2 heures (de 10h30 à 12h30) 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 clairement présentées.**

1._____/15 (1.1 ; 1.2 ; 1.3 ; 1.4 ; 1.5)

2._____/15 (2.1 ; 2.2 ; 2.3 ; 2.4 ; 2.5 ; 2.6 ; 2.7 ; 2.8)

3._____/20 (3.1 ; 3.2 ; 3.3 ; 3.4)

4._____/20 (4.1)

5._____/30 (5.1)

Total:_____/100

Question 1 (15 points)

1.1 Où sont fixées les valeurs des paramètres par défaut? Dans la déclaration ou dans la définition des fonctions ? Pourquoi ?

1.2 Quelles différences y a-t-il entre un passage par valeur, un passage par adresse et un passage par référence pour la valeur de retour d'une fonction?

1.3 Quand passe-t-on un paramètre par référence ? Quand passe-t-on un paramètre par référence constante?

1.4 Pourquoi faut-il préférer le compilateur au préprocesseur ?

1.5 Quelle différence y a-t-il entre une copie profonde et une copie superficielle ?

Question 2 (15 points) Soit le fragment de code suivant:

```
namespace Noir {
      int j;
      void print(int);
      char ch;
}
namespace Blanc {
      int j;
      void print(int);
      double vision;
}

int j;

void UneFonction() {
      using namespace Noir;
      using namespace Blanc;
      j = 0;            // -1-
      print(5);         // -2-
      ch = 'a';         // -3-
      vision = 7.65;    // -4-
      int j = 10;       // -5-
      ::j = 5;          // -6-
      Blanc::j = 5;     // -7-
      Noir::print(j);   // -8-
}
```

Tout en justifiant votre réponse, dites si les appels -1- à -8- sont corrects ou pas ? (Chaque question est indépendante.)

2.1 `j = 0;`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.2 `print(5);`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.3 `ch = 'a';`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.4 `vision = 7.65;`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.5 `int j = 10;`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.6 `::j = 5;`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.7 `Blanc::j = 5;`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

2.8 `Noir::print(j);`

Correct ☐ **Incorrect** ☐

Pourquoi? (Courte explication)

Question 3 (20 points)

3.1 Tout en justifiant votre réponse, que va afficher en sortie le programme suivant, qui compile et s'exécute correctement.

```
#include <iostream>
void sosie(int) ;
void sosie(double) ;

using namespace std ;

int main() {
      int n = 5;
      double x = 2.5 ;
      sosie(n) ;
      sosie(x) ;
      return 0 ;
}
void sosie (int a) { // sosie -1-
      cout << "sosie 1 a = " << a << endl;
}
void sosie (double a) { // sosie -2-
      cout << "sosie 2 a = " << a << endl;
}
```

3.2 Tout en justifiant votre réponse, quelle fonction sosie, du précédent programme, est appelée dans les cas suivants ?

```
char c ; float y ;
```

```
sosie (c);
```

sosie -1- ☐

sosie -2- ☐

```
sosie (y);
```

sosie -1- ☐

sosie -2- ☐

```
sosie ('d');
```

sosie -1- ☐

sosie -2- ☐

3.3 Quelle fonction est appelée avec ces déclarations ?

```
void affiche (char*) ; // affiche -1-
void affiche (void*) ; // affiche -2-
char *py ; double *px ;
```

```
affiche (py);
```

affiche -1- ☐

affiche -2- ☐

```
affiche (px);
```

affiche -1- ☐

affiche -2- ☐

3.4 Quelle fonction est appelée avec ces déclarations ?

```
void choix (int &) ; // choix -1-
void choix (const int &) ; // choix -2-
int n=3 ; const int p=5 ;
```

```
choix (n);
```

choix -1- ☐

choix -2- ☐

```
choix (p);
```

choix -1- ☐

choix -2- ☐

```
choix (12);
```

choix -1- ☐

choix -2- ☐

Question 4 (20 points)

Cette classe contient des erreurs de syntaxe et des avertissements ("warnings"). Indiquez-les et précisez les, puis proposez une correction possible. Les numéros de ligne qui figurent au début de chaque ligne ne font pas partie du code et servent à identifier clairement les erreurs.

```
 1  class VivementCeSoir {

 2      int i = 19;
 3      int k;
 4      int j = 15.2;

 5  public
 6      int uneFonction (int i) {
 7          int j;
 8          k = i;
 9          if (k<2) return 3;
10          cout << "j=" << j << endl;
11      }

12      int uneAutre () {
13          cout << "k=" << k << endl;
14          int i = 15;
15          if (i>15) return 5;
16          else {
17              for (int i=0; i<15; i++) k++;
18          }
19      }
20  }
```

Question 5 (30 points)

Cet exercice a pour but l'implantation d'un type abstrait Ensemble en C++. Vous allez pour cela concevoir quelques méthodes de la **classe Ensemble** et **un (ou plusieurs) fragment de programme** permettant de tester ces méthodes.
Les seuls ensembles considérés ici sont des ensembles d'entiers. Un ensemble est représenté par un tableau d'entiers (ensemble) dont le nombre maximal d'éléments est spécifié par l'utilisateur de la classe au moment de la création d'un objet de type Ensemble. Par exemple l'ensemble {10, 1234, 56} est représenté par une table d'entiers dont le premier élément (indice 0) est 10, le deuxième élément (indice 1) est 1234 et le troisième élément (indice 2) est 56.

Il vous est demandé de concevoir les méthodes suivantes:

- boolean appartient(int element): Cette méthode retourne vrai si l'élément passé en argument (ici element) appartient à l'ensemble; faux sinon.

- int getSize(): Cette méthode retourne le nombre d'éléments contenus dans l'ensemble.

- void ajouter(int element): Cette méthode permet d'ajouter l'élément (element) dans l'ensemble s'il n'existe pas déjà.

- Ensemble union(Ensemble e): Cette méthode crée un nouvel ensemble constitué de l'union de l'ensemble «appellant » avec celui passé en argument.

Vous pouvez ajouter autant de données et de méthodes membres que vous le souhaitez.

Attention : Il ne vous est pas permis d'utiliser les fonctionnalités de la bibliothèque STL.