Université Djillali Liabes de Sidi-Bel-Abbes
Faculté de génie électrique
Département de Télécommunications

1ère année Master RT
Semestre 1
Janvier 2018

Examen de Programmation orientée Objets en C++

Durée : 1h30

EXERCICE 1 : Questions de cours (5 points)

1. Quel est le principal avantage pratique de la programmation Orientée Objets ?
2. Quel est le principal objectif de l'héritage ?
3. Comment organiser un code Orientée Objets ?
4. A quoi sert un constructeur d'une classe ?
5. Comment peut-on différencier un constructeur?

EXERCICE 2 : (5 points)

1. Quels seront les résultats fournis par ce programme :

```
#include <iostream>
using namespace std ;
class A
        { int na ;
        public :
        A (int nn=1)
        { na = nn ; cout << "$$construction objet A " << na << "\n" ;
        }
} ;
class B : public A
        { float xb ;
        public :
        B (float xx=0.0)
        { xb = xx ; cout << "$$construction objet B " << xb << "\n" ;
        }
} ;
class C : public A
        { int nc ;
        public :
        C (int nn= 2) : A (2*nn+1)
        { nc = nn ;
        cout << "$$construction objet C " << nc << "\n" ;
        }
} ;
class D : public B, public C
        { int nd ;
        public :
        D (int n1, int n2,float x) : C (n1), B (x)
        { nd = n2 ;
        cout << "$$construction objet D " << nd << "\n" ;
        }
} ;
main()
        { D d (10, 20, 5.0) ;
        }
```

2. Transformer le programme précédent, de manière qu'un objet de type D ne contienne qu'une seule fois les membres de A. On s'arrangera pour que le constructeur de A soit appelé avec la valeur 2***nn***+1, dans laquelle **nn** désigne l'argument du constructeur de C.

EXERCICE 3 : (5 points)

Voici le texte d'une classe représentant un compte bancaire et les opérations bancaires courantes.

```
class Compte{
        int solde = 0;
        public:
                void deposer(int montant){
                        solde = solde + montant;
                        }
                void retirer(int montant){
                         solde = solde -montant;
                        }
                void virerVers(int montant, Compte &destination){
                         retirer(montant);
                        destination.deposer(montant);
                         }
                void afficher(){
                        cout<<"solde: "<<solde<<"\n";
                }
};
```

a. Comment fonctionne la méthode **virerVers** ? Combien de comptes fait-elle intervenir ?

b. Créez deux comptes que vous affecterez à deux variables. Ecrivez le code correspondant aux opérations suivantes :

- dépôt de 500 dinars sur le premier compte.
- dépôt de 1000 dinars sur le second compte.
- retrait de 10 dinars sur le second compte.
- virement de 75 dinars du premier compte vers le second.
- affichage des soldes des deux comptes.

Mettez le code correspondant à cette question dans la fonction main.

EXERCICE 4 : (5 points)

Calculer la somme des n premiers termes de la série harmonique, c'est-à-dire la somme :

$1 + 1/2 + 1/3 + 1/4 + ... + 1/n$

La valeur de n sera lue en donnée.

Université Djillali Liabes de Sidi-Bel-Abbes
Faculté de génie électrique
Département de Télécommunications

1ère année Master RT
Semestre 1
Janvier 2018

Correction de l'Examen de Programmation Orientée Objets en C++

EXERCICE 1 : Questions de cours (5 points)

1. Quel est le principal avantage pratique de la programmation Orientée Objets ?
   Le découpage d'un programme complexe en des parties plus simples **1 Pt**
2. Quel est le principal objectif de l'héritage ?
   La réutilisation de code existant (grâce au principe de sous-classes). **1 Pt**
3. Comment organiser un code Orientée Objets ?
   Une classe par fichier **1 Pt**
4. A quoi sert un constructeur d'une classe ?
   A regrouper une série d'opération qui sera répétée pour chaque objet créé (par exemple: initialiser plusieurs variables). **1 Pt**
5. Comment peut-on différencier un constructeur?
   Grâce au fait qu'il n'ait aucun type de retour (même pas de void) **1 Pt**

EXERCICE 2 : (5 points)

1. Voici les résultats obtenus :

```
$$construction objet A 1
$$construction objet B 5
$$construction objet A 21
$$construction objet C 10
$$construction objet D 20
```

0.5 Pt x 5=2.5 Pts

2. En résumé, la déclaration de A reste inchangée, celle de B est transformée en :

```
class B : public virtual A
{ // le reste est inchangé
}
```

0.75 Pt

Celle de C est transformée de façon analogue :

```
class C : public virtual A
{ // le reste est inchangé
}
```

0.75 Pt

Enfin, dans D, l'en-tête du constructeur devient :

```
D (int n1, int n2, float x) : C (n1), B (x), A (2*n1+1)
```

1 Pt

À titre indicatif, voici les résultats que fournirait le programme précédent ainsi transformé :

```
$$construction objet A 21
$$construction objet B 5
$$construction objet C 10
$$construction objet D 20
```

EXERCICE 3 : (5 points)

a. Comment fonctionne la méthode **virerVers** ? Combien de comptes fait-elle intervenir ?
La méthode virerVers fait intervenir deux objets de type Compte : l'objet sur lequel la méthode est appelée et destination, le paramètre de la méthode. Le virement s'effectue vers le paramètre de la méthode. L'argent est retiré d'un compte et déposé sur l'autre. **1 Pt**

b.

```
int main()
{
Compte karime, fethi;
// dépôt de 500 dinars sur le premier compte.
karime.deposer(500);
// dépôt de 1000 dinars sur le second compte.
```

0.5 Pt **0.5 Pt** **0.5 Pt**

```
        fethi.deposer(1000);
        // retrait de 10 dinars sur le second compte. 0.5 Pt
        fethi.retirer(10);
        // virement de 75 dinars du premier compte vers le second.  1 Pt
        karime.virerVers(75, fethi);
        // affichage des soldes des deux comptes.  0.5 Pt
        cout<<"Compte de karime, ";
        karime.afficher();
        cout<<"Compte de fethi, ";
        fethi.afficher();
        return 0;
}
```

EXERCICE 4 : (5 points)

Calculer la somme des n premiers termes de la série harmonique, c'est-à-dire la somme :

1 + 1/2 + 1/3 + 1/4 + ... + 1/n

```
#include <iostream>
using namespace std ;
main()
        {
                int nt ; /* nombre de termes de la série harmonique */
                float som ; /* pour la somme de la série */
                int i ;
                do
                        { cout << "combien de termes : " ;
                        cin >> nt ;
                        }
                while (nt<1) ;
                for (i=1, som=0 ; i<=nt ; i++) som += (float)1/i ;
                cout << "Somme des " << nt << " premiers termes = "<< som ;
}
```