INFO-F-202
Langages de programmation 2
Université libre de Bruxelles

Examen

Chaque question vaut 4/20.

1. Écrivez du code pour une classe appelée `Rectangle` qui fonctionnera comme illustré dans cet exemple. Notez que `pair` est une classe qui fait partie de la bibliothèque standard C++ ; il stocke simplement deux éléments des types donnés dans des variables membres publiques appelées `first` et `second`.

```
Rectangle r{2,10};
cout<<r.area()<<endl;
pair<double,double> p{5,10};
Rectangle r2{p};
string s{r};
cout<<s<<endl;
p=r;
cout<<p.first<<endl;
```

```
20
[2.000000,10.000000]
2
```

2. En Python, vous pouvez écrire le code suivant :

```python
def transform(A,f):
  return [f(x) for x in A]

def square(x):
  return x*x


V1=[1,2,3,4]
V2=transform(V1,square)
print(V2)

V3=["1","2","3","4","5"]
V4=transform(V3,lambda x:"0x"+x)
print(V4)
```

Il affichera ce qui suit :

```
[1, 4, 9, 16]
['0x1', '0x2', '0x3', '0x4', '0x5']
```

Afficher le code pour savoir comment procéder en C++. Vous devez coder `transform`, `square` et le code de démonstration. Le premier paramètre de `transform` et sa valeur de retour doivent être `vector`.

3. Écrivez du code pour définir quelque chose appelé `Range` qui fonctionne comme en Python. Par exemple, cela devrait fonctionner comme suit :

```
for (auto x:Range(5,10)) cout<<x<<" "; cout<<endl;
for (auto x:Range(10)) cout<<x<<" "; cout<<endl;
for (auto x:Range(100,1000,100)) cout<<x<<" "; cout<<endl;

int sum=0;
for (auto x:Range(1000000)) sum+=x;
cout<<sum<<endl;
```

```
5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
100 200 300 400 500 600 700 800 900
1783293664
```

Vous pouvez supposer que seuls des entiers non négatifs sont donnés à `Range`. Vous ne devez utiliser aucun type de tableau ou `vector` dans votre code `Range`.

4. Votre tâche consiste à écrire du code pouvant représenter des expressions avec une seule variable x et à les évaluer.

   Par exemple, pour représenter $x + 5x$ et imprimer ce qu'il évalue pour x de 0 à 9, vous pouvez écrire :

```
auto E=new Sum(new X(),new Product(new Number(5),new X()));
for (auto x:Range(10)) // Range est défini dans problème 3
  cout<<x<<" "<<E->evaluate(x)<<endl;
```

   Ceci imprime :

```
0 0
1 6
2 12
3 18
4 24
5 30
6 36
7 42
8 48
9 54
```

   Comme vous pouvez le voir, il existe quatre classes, `Sum`, `Product`, `Number` et `X`. Chacune doit avoir une méthode appelée `evaluate` et avoir un constructeur qui fonctionne comme dans l'exemple et qui, ensemble, peuvent être utilisés pour construire n'importe quelle expression. Écrivez le code nécessaire pour ce faire. Votre code ne doit pas utiliser de pointeurs intelligents, ne pas avoir de fuites de mémoire et prendre en charge la construction et l'assignation de copie et de déplacement. Vous n'avez pas besoin d'écrire le code pour `Product`, je supposerai qu'il est identique à `Sum` avec un $*$ au lieu d'un $+$ à l'endroit approprié.

5. Considérez le code suivant :

```
1  class Count {
2    static int count;
3    int myCount=0;
4  public:
5    Count() {myCount=count++;}
6    int get() const {return myCount;}
7  };
8  int Count::count=0;
9
10
11 class Point : public Count {
12   int x,y;
13 public:
14   Point(int x,int y):x{x},y{y}{};
15   string description() const {return to_string(x)+" "+to_string(y);}
16 };
17
18 class Color : public Count {
19   int r,g,b;
20 public:
21   Color(int r,int g,int b):r{r},g{g},b{b}{}
22   string description() const {return to_string(r)+" "+to_string(g)+" "+to_string(b);}
23 };
24
25 class ColoredPoint : public Point, public Color {
26 public:
27   ColoredPoint(int x,int y,int r,int g,int b):Point{x,y},Color{r,g,b}{}
28   string description() const {return Point::description()+" "+Color::description();}
29 };
30
31 void demo5(){
32   ColoredPoint cp{1,2,3,4,5};
33   cout<<cp.get();
34   cout<<cp.Point::get()<<endl;
35   cout<<cp.Color::get()<<endl;
36   cout<<cp.Count::get()<<endl;
37   cout<<cp.description()<<endl;
38   Point *p=&cp;
39   cout<<p->description()<<endl;
40 }
```

Pour chaque instruction d'impression dans `demo5`, indiquez soit ce qu'elle affiche, soit qu'il s'agit d'une erreur et qu'elle ne sera pas compilée.

— Ligne 33 : `cout<<cp.get();`

— Ligne 34 : `cout<<cp.Point::get()<<endl;`

— Ligne 35 : `cout<<cp.Color::get()<<endl;`

— Ligne 36 : `cout<<cp.Count::get()<<endl;`

— Ligne 37 : `cout<<cp.description()<<endl;`

— Ligne 39 : `cout<<p->description()<<endl;`