université PARIS DIDEROT PARIS 7

EXAMEN Session 2 | Master 1 Informatique – LOA | 22/6/2015 8^{30}–11^{30}

Aucun document ou support autre que le sujet ou les copies d'examen n'est autorisé.
(La copie ou les brouillons du voisin ne sont pas des supports autorisés).
Éteignez impérativement vos mobiles.

1 Exercice

On souhaite construire un type (`MyInt`) dont le comportement est exactement celui d'un entier (`int`), mais qui permette en plus d'effectuer des mesures sur l'ensemble des opérations arithmétiques réalisées avec celui-ci.

1. Écrire un prototype de ce type de sorte que l'on puisse compiler le programme suivant :

```
int main() {
  MyInt i=2, j=5;
  const MyInt _3(3);
  const MyInt _2(2);

  MyInt k = _2*i+j/_3+_2;
  int l = k;
  cout << k << "==" << l << endl;
  MyInt::showStats();
}
```

   et que le résultat produit à l'exécution soit exactement :

```
7==7
2 additions, 1 multiplications, 1 divisions
```

2. Combien d'objets de type `MyInt` sont-ils créés à l'exécution du programme ? Comment vous en convaincre ?

2 Exercice

Soit le code (incomplet) suivant :

```
template <typename E,int SIZE,int DIM> class Array {
  Array<E,SIZE,DIM-1> array[SIZE];
public:
  Array<E,SIZE,DIM-1> &get(int n) { return array[n]; }
  const Array<E,SIZE,DIM-1> &get(int n) const { return array[n]; }
};

template <typename E,int SIZE> class Array<E,SIZE,1> {
  E elements[SIZE];
public:
  Array() { for (int i=0; i<SIZE; i++) elements[i]=E(); }
  E &get(int n) { return elements[n]; }
  const E &get(int n) const { return elements[n]; }
};

int main() {
  Array<int,5,2> a;
  cout << a.get(0).get(3) << endl;
  a.get(0).get(3) = 44;
  cout << a.get(0).get(3) << endl;
  a[0][3] = 12;
  cout << a.get(0).get(3) << endl;
  cout << a << endl;
}
```

1. Quelles sont les lignes du `main` qui provoquent une erreur de compilation ? Expliquez pourquoi...
2. Quelles sont les méthodes de quels types qui sont appelées à l'exécution de l'instruction `a.get(0).get(3)` ?
3. Modifier les définitions des *templates* de sorte que le code compile et s'exécute en fournissant le résultat suivant :

```
0
44
12
0,0,0,12,0
0,0,0,0,0
0,0,0,0,0
0,0,0,0,0
0,0,0,0,0
```

3 Exercice

On souhaite simuler la gestion d'un parking de véhicules (en souterrain). Les véhicules qui sont autorisés à s'y rendre pour se garer sont : les voitures, les motos, les scooters, les camionnettes de moins de 5 tonnes et moins de 5 mètres de long et moins de 2 mètres de large. Chaque véhicule pourra être accueilli sur n'importe qu'elle place disponible (il n'y a pas de place réservée à une catégorie ou une autre). Le parking possède 200 places. De plus, le parking est payant : 1€ de l'heure. Les véhicules sont identifiés par leur plaque minéralogique (un simple numéro de série attribué à la création de tout véhicule par le service des mines). Le paiement s'effectue en contactant les services fiscaux dès qu'un véhicule sort du parking après avoir calculé le délai écoulé entre son entrée et sa sortie (toute heure commencée est due intégralement).

1. Établissez un diagramme UML du problème en faisant apparaître tous les concepts utiles, les relations entre concepts en faisant apparaître si nécessaire les arités et les rôles. Ne pas oublier les attributs et opérations les plus utiles.
2. Écrivez la déclaration des classes.
3. Écrivez la définition complète des méthodes qui interviennent dans un scénario qui ferait entrer une voiture dans le parking, attendrait 2h30 puis ferait sortir la voiture.

4 Exercice

```
1  class Premiere {
2  public: Premiere()             { cout << "ctor␣premiere" << endl; }
3          virtual ~Premiere()    { cout << "dtor␣premiere" << endl; }
4          void salut()           { cout << "salut" << endl; }
5          virtual void bonjour() { cout << "bonjour" << endl; }
6  };
7  class Seconde : public Premiere {
8  public: Seconde()              { cout << "ctor␣seconde" << endl; }
9          virtual ~Seconde()     { cout << "dtor␣seconde" << endl; }
10         void salut()           { cout << "salutations" << endl; }
11         virtual void bonjour() { cout << "bien␣le␣bonjour" << endl; }
12 };
13 int main() {
14   Premiere p;
15   Seconde *s = new Seconde;
16   p.salut(); p.bonjour();
17   (*s).salut(); (*s).bonjour();
18   Seconde *ps = s;
19   ps->salut(); ps->bonjour();
20   Premiere *pp = ps;
21   pp->salut(); pp->bonjour();
22   delete s;
23   return 0;
24 }
```

1. Indiquez, en argumentant, quels sont les affichages produits à l'exécution du code précédent.