NOM: **PRENOM:**

Année 2018-2019 2ème session

PROGRAMMATION C++
PG212
JULIEN ALLALI

Filière : informatique Année : 2 Semestre : S7

Date de l'examen : 8 Mars 2019 Durée de l'examen : 2h

Documents autorisés ☐ sans document ☒
Calculatrice autorisée ☐ non autorisée ☒

Autre : **Les réponses aux questions doivent se faire sur le sujet d'examen, écrire proprement et lisiblement**

SUJET

Commencer par écrire votre nom et prénom sur chaque feuille.
Attention: tous les détails compteront dans la notation: **clarté & lisibilité**, *factorisation, nommage, validité!*

▸**Exercice 1.** *Stock : /10*

Vous devez développer un système permettant de gérer un stock d'objets. Chaque objet possède les propriétés: référence (un entier), poids (en kg), hauteur (en m), largeur (en m) et longueur (en m). De plus certains objets peuvent contenir d'autres objets (c'est le cas d'un carton pas exemple).

L'interface `Object` *permet de connaître les propriétés de l'objet et de savoir s'il peut en contenir d'autres via une méthode* isContainer. *Cette interface possède également une méthode* asContainer *qui retour* la vue *conteneur de l'objet si* isContainer *revoie vrai et renvoie NULL sinon.*

L'interface `ObjectContainer` *(qui est un* `Object`*) permet d'ajouter un objet (*add*) , savoir si un objet est présent (*has*) et supprimer un objet (*remove*). Elle possède également la méthode* getObjects *qui renvoie une liste des identifiants des objets contenus.*

Voici un exemple d'utilisation de l'ensemble des interfaces:

```
1  #include <vector>
2
3  int main(){
4    Object *a=new Ball(10);
5    Object *b=new Skate(11);
6    Object *box=new Box(100,100,100,12); // hauteur, largeur, longueur, id
7    Container *c=box->asContainer();
8    c->reference();
9    c->add(a);
10   c->add(b);
11   c->has(10); // renvoi vrai
12   Object *o=c->remove(10); // renvoi l'adresse de l'instance Ball
13   c->has(10); // renvoie faux
14   c->remove(10); // renvoie NULL
15   std::vector<int > x=c->getObjects();
16   x.size(); // le nombre d'object contenu
```

NOM: **PRENOM:**

```
17   for(int i=0;i<x.size();++i){
18     Object *p=c->get(x[i]); // x[i] est l'identifiant d'un objet
19     p->reference(); // doit etre egale a x[i]
20     std::cout<<"The container "<<c->reference()<<" contains "<<p->reference();
21     std::cout<< " of size "<<p->width()<<"x"<<p->height()<<"x"<<p->depth()<<std::endl;
22   }
23   // Affiche: The container 12 contains 11 of size 0.8x0.15x0x2
24   delete a;
25   delete b;
26   delete box;
```

Attention: *tous les détails compteront dans la notation:* **clarté & lisibilité**, *factorisation, nommage, validité!*

Headers: *Pour simplifier, vous écrirez l'implémentation des méthodes lors de la déclaration de celle-ci (pas de séparation .hpp/.cpp)*

Donner le code des interfaces `Object` *et* `Container`. *Vous veillerez à faire en sorte que votre code soit compatible avec le code donné en exemple :*

/5

NOM: **PRENOM:**

Donner le code d'une classe SimpleObject *implémentant l'interface* Object *mais qui ne puisse pas contenir d'objets :*

/2

NOM: **PRENOM:**

Donner le code d'une fonction recursivePrint *qui prend une instance d'objet en paramètre et qui affiche les informations sur celui-ci. Si l'objet est un conteneur, alors son contenu doit être également affiché (et ce recursivement).*

/3

NOM: **PRENOM:**

▸ **Exercice 2.** *Opérateurs : /3*

Donner le code d'une classe `Entier` *telle que le code suivant puisse fonctionner :*

```
1
2  void printInt(int i){
3    printf("%d\n",i);
4  }
5
6  int main(){
7    Entier a=1;
8    Entier b(3);
9    Entier c=10;
10   a=b+c;
11   printInt(a);
12 }
```

/3

NOM: **PRENOM:**

▸ **Exercice 3.** *Cours : /7*

Quelles sont les caratéristiques d'une classe implémentant le concept object de classe abstraite *en C++?*

/1.5

Quelles doivent être les caratéristiques d'une classe implémentant le concept object d'interface *en C++?*

/1.5

Si une variable i est ni un pointeur, ni une référence mais bien une instance de la classe C qui contient une méthode m virtuelle, que pouvez-vous dire de l'appel `i.m()` *:*

/1

NOM: **PRENOM:**

Je dispose d'une référence constante i vers une instance de la classe `A`*, quelles méthodes puis-je appeler sur i? Quand est-il si la variable i n'est pas constante?*

/2

La classe A possède un unique constructeur qui prend un entier. Donner le code d'une classe B dérivée de A et qui soit instanciable:

/1