**BQ1000. Class Set**

**Description**

设计一个集合(Set)类模板，Set可以包含**零个**或**多个无序的非重复**元素。

公共接口包含以下成员函数：

1. 创建一个Set；

2. 增加一个新元素到Set；

3.从Set中删除一个元素

4.列举Set中的所有元素

5. s1和s2都是Set的对象，计算它们的并集（s1 + s2）= { x | x 属于 s1 或 x 属于 s2 }

6. s1和s2都是Set的对象，计算它们的差集（s1 - s2 ) = { x | x 属于 s1 且 x 不属于 s2 }

该类模板的声明如下：

**template**< **typename** T >  
**class** Set  
{  
**public**:  
 Set() ;  
 Set( **int** maxSize ) ;  
 ~Set() ;  
 Set( **const** Set& ) ;  
 Set **operator**+( **const** Set& s ) ;  
 Set **operator**-( **const** Set& s ) ;  
 **void operator**=( **const** Set& s ) ;  
 **bool** insert( T newItem ) ;  
 **bool** erase( T targetItem ) ;  
 **void** print() **const** ;  
 T\* getData() **const** ;  
 **int** getCount() **const** ;  
 **int** getMaxSize() **const** ;  
**private**:  
 **int** count ;  
 **int** maxSize ;  
 T\* data ;  
} ;

**Input**

 Set<**int**> s1( 10 ), s2( 10 ), s\_plus, s\_minus ;  
   
 s1.insert( 1 ) ;  
 s1.insert( 2 ) ;  
 s1.insert( 2 ) ;  
 s1.insert( 4 ) ;  
 cout << "Items in s1 are : " ;  
 s1.print() ;  
   
 s2.insert( 3 ) ;  
 s2.insert( 3 ) ;  
 s2.insert( 1 ) ;  
 s2.insert( 2 ) ;  
 cout << "Items in s2 are: " ;  
 s2.print() ;  
   
 s\_plus = s1 + s2 ;  
 cout << "Items in s\_plus are: " ;  
 s\_plus.print() ;  
   
 s\_minus = s1 - s2 ;  
 cout << "Items in s\_minus are: " ;  
 s\_minus.print() ;

**Output**

Items in s1 are : 1 2 4   
Items in s2 are: 3 1 2   
Items in s\_plus are: 1 2 4 3   
Items in s\_minus are: 4

**Hint**

提交的代码中，请不要包含main()函数

**BQ1001. Set**

**Description**

Please implement the function int sumOfIntersection(const set<int>& set1, const set<int>& set2), which returns the sum of all the elements that both appear in set1 and set2.

Your submitted source code should include the implementation of the sumOfIntersection function.

No main() function should be included.

**Sample Input**

http://soj.sysu.edu.cn/images/clipboard.jpgCopy sample input to clipboard

set1: 1 3 5

set2: 1

**Sample Output**

1

**BQ1002. 可排序数组**

**Description**

可排序数组 SortableArray定义如下：

1. 可通过成员函数pushBack插入一个元素

2.可通过成员函数sort对数组进行升序排序

3.可通过成员函printArray数输出数组

4.可排序数组SortableArray是一个模板类

5.SortableArray的实例：SortableArray<int> array( maxArraySize ) ;

（请实现SortableArray，注意在提交的代码中请不要包含 main()函数 ）

**Input**

测试框架如下：

int maxSize = 10 ;

SortableArray< int > arr( maxSize ) ;

arr.pushBack( 45 ) ;

arr.pushBack( 32 ) ;

arr.pushBack( 41 ) ;

arr.printArray() ;

arr.sort() ;

arr.printfArray() ;

**Output**

45 32 41

32 41 45

**BQ1003. Point & Vector**

**Description**

平面上有好多点，男神叫py去挪他们。用什么挪呢？男神给了py一根向量。

这本来也不是什么麻烦事，可是男神比较DT，他很有艺术细胞，他喜欢美的表达。男神提出了这么几个要求：

（假定有Vector a, b; Point c;）

1.可以直接在输出流里能输出点和向量，形如cout << a;输出格式为x坐标与y坐标之间以一个空格隔开，在输出运算符重载的实现里不要换行。

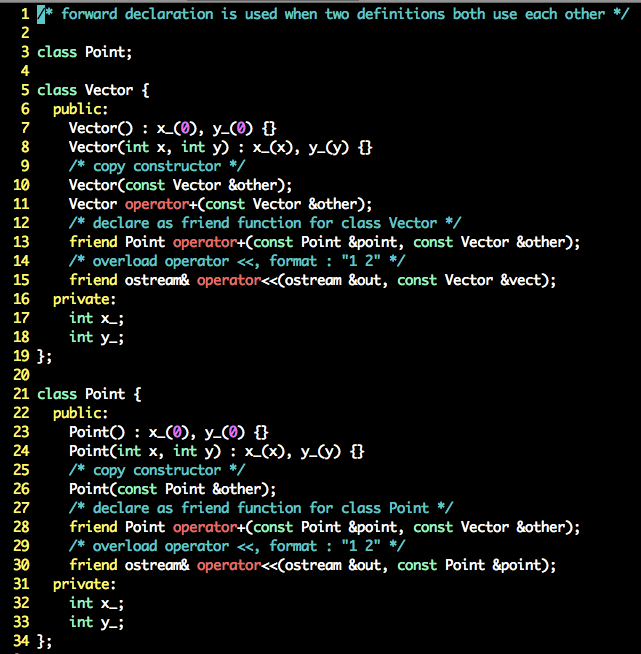
2.重载Vector与Vector的加法，形如a = a + b;

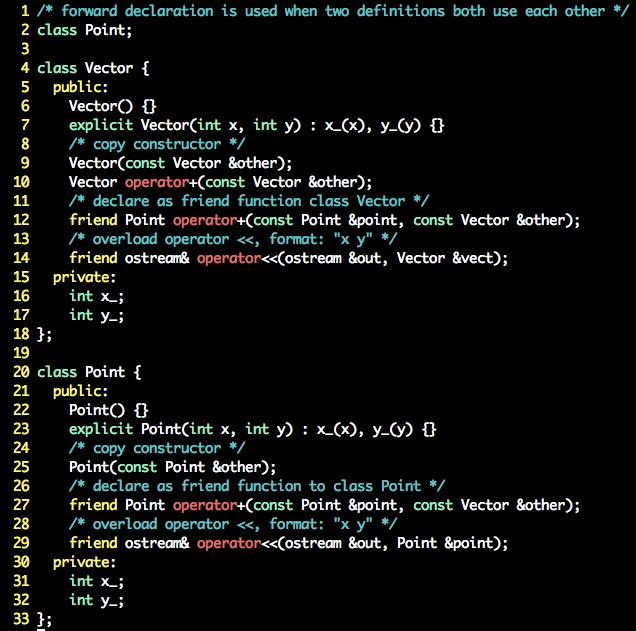
3.重载Point与Vector的加法，即将点沿着该向量位移。形如c = c + a;

男神说，你不觉得这样很优雅吗？

虽然py不觉得，但他没什么所谓，因为写程序的人是你。

好心的py给了你类的定义。老规矩，提交类的定义和实现，不提交main函数。





**Hint**

关于forward-declaration在stackoverflow（著名编程问答网站，别再百度知道了）有一个[很不错的问题](http://stackoverflow.com/questions/4757565/c-forward-declaration)

**BQ1004. C++:Instance of**

**Description**

在Java语言里边，所有的对象都会继承Object类，但C++却是没有的

例如以下的一些类：

class Object;

class Animal:public Object;

class Dog:public Animal;

class Cat:public Animal;

class Vehicle:public Object;

class Bus:public Vehicle;

class Car:public Vehicle;

class Person:public Object;

class Student:public Animal;

class Teacher:public Animal;

===============================================================

我们知道对象是可以向上向下转型的，就例如

Animal\* animal = new Dog();

因为继承本身就是一种is-a的关系，狗就是动物，因此这样赋值是没问题的。

但是问题来了。倘若别人传递上述代码的animal对象给你，你这个时候却不知道它原本是Dog这个类的（在Java里边是有instance of 可以判断的），于是现在你的任务来了。

请完成以下函数以及类，使得满足instanceof函数可以返回该对象的类型名字。（注意这里的类全部都还没定义，你可以根据自己的思路去完成定义，完成的方法不限，能AC即可）

================================================================

class Object;

class Animal:public Object;

class Dog:public Animal;

class Cat:public Animal;

class Vehicle:public Object;

class Bus:public Vehicle;

class Car:public Vehicle;

class Person:public Object;

class Student:public Animal;

class Teacher:public Animal;

/\*

 \* 一个Object对象

 \* 返回这个对象实例实际的类型名

 \* 例如： Object \*obj = new Bus();

 \* 则instanceof(\*obj) == "Bus"

 \*/

string instanceof(Object obj)

**Input**

 N/A

**Output**

 N/A

**Sample Input**

http://soj.sysu.edu.cn/images/clipboard.jpgCopy sample input to clipboard

样例测试:

Object \*obj = new Dog();

cout << instanceof(\*obj) << endl;

Object \*obj = new Car();

cout << instanceof(\*obj) << endl;

**Sample Output**

Dog

Car

**Hint**

 无需提交主函数！

**BQ1005. Template Specification**

**Description**

给定一个函数声明如下 bool cmp( T a, T b )。

该函数比较变量 a 和 b 大小，如果a等于b，返回true；否则返回false。

        （1） 如果a, b的类型是 int, float, double, char\* 或 string 函数cmp 比较的是 a, b 值的大小；

                 （注意char\*字符串的比较请使用函数strcmp（头文件string.h））

        （2）如果a, b的类型是指针，则比较的是其所指向的对象是否相等。

**Input**

 int aInt = 1, bInt = 2 ;  
 double aDouble = 3.0, bDouble = 3.0 ;  
 char aChars[5] = "haha", bChars[5] ;  
 string aStr = "haha", bStr = "aha" ;  
 int\* aIntPtr = &aInt, \*bIntPtr = &bInt ;

 cout << boolalpha << cmp( aInt, bInt ) << endl ;  
 cout << cmp( aDouble, bDouble ) << endl ;

 strcpy( bChars, aChars ) ;  
 cout << cmp( aChars, bChars ) << endl ;  
 cout << cmp( aStr, bStr ) << endl ;  
 cout << cmp( aIntPtr, bIntPtr ) << endl ;

**Output**

false  
true  
true  
false  
false