Jorge A. Serrano

Jewelry Box modulo 10

Mi duda es en el main, como llamo el cistream operator de los punteros. EL ostream funciona bien pero no me deja usar el istream que tengo adjuntado ahora miismo.

#ifndef \_BOX\_INTERFACE\_

#define \_BOX\_INTERFACE\_

#include<iostream>

using namespace std;

template < class ItemType>

class BoxInterface {

public:

virtual ~BoxInterface() {}

virtual void setItem(const ItemType& theItem) = 0;

virtual ItemType getItem() const = 0;

};

#endif

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#pragma once

#include<iostream>

#include<string>

using namespace std;

class Jewelry

{

private:

char\* gender;

char\* type;

int weight;

double price;

char\* metal;

public:

Jewelry();// Default Constructor

Jewelry(char\* aGender, char\* aType, int aWeight, double aPrice, char\* aMetal);// Constructor with parameters

Jewelry(const Jewelry&);// Copy Constructor

~Jewelry();// Destructor

// Mutators

void setGender(char\* aGender);

void setType(char\* aType);

void setWeight(int aWeight);

void setPrice(double aPrice);

void setMetal(char\* aMetal);

// Accessors

char\* getGender() const;

char\* getType() const;

int getWeight() const;

double getPrice() const;

char\* getMetal() const;

friend ostream& operator<<(ostream&, const Jewelry&);

friend istream& operator>>(istream&, Jewelry&);

};

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#ifndef \_JEWELRY\_BOX

#define \_JEWELRY\_BOX

#include "PlainBox.h"

template <class ItemType>

class JewelryBox : public PlainBox<ItemType>

{

private:

bool firstItemStored;

public:

JewelryBox();

JewelryBox(const ItemType& aItem);

~JewelryBox();

void setItem(const ItemType& aItem);

};

template<class ItemType>

JewelryBox<ItemType>::JewelryBox() :PlainBox<ItemType>() {

firstItemStored = false;

}

template<class ItemType>

JewelryBox<ItemType>::JewelryBox(const ItemType& aItem) : PlainBox<ItemType>() {

PlainBox<ItemType> ::setItem(aItem);

firstItemStored = false;

}

template<class ItemType>

JewelryBox<ItemType>::~JewelryBox() {

cout << "JewelryBox destructor executing\n";

}

template<class ItemType>

void JewelryBox<ItemType>::setItem(const ItemType& aItem) {

if (!firstItemStored) {

PlainBox<ItemType> ::setItem(aItem);

firstItemStored = true;

}

}

#endif

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#ifndef \_PLAINT\_BOX

#define \_PLAINT\_BOX

#include "BoxInterface.h"

#include<iostream>

using namespace std;

template<class ItemType>

class PlainBox : public BoxInterface<ItemType> {

private:

ItemType item;

public:

PlainBox();

PlainBox(const ItemType& theItem);

virtual~PlainBox();

virtual void setItem(const ItemType& theItem);

virtual ItemType getItem() const;

};

template<class ItemType>

PlainBox<ItemType>::PlainBox() {

cout << "PlainBox constructor executing\n";

}

template<class ItemType>

PlainBox<ItemType>::PlainBox(const ItemType& theItem) {

setItem(theItem);

}

template<class ItemType>

PlainBox<ItemType>::~PlainBox() {

cout << "PlainBox destructor executing\n";

}

template<class ItemType>

void PlainBox<ItemType>::setItem(const ItemType& theItem) {

item = theItem;

}

template<class ItemType>

ItemType PlainBox<ItemType>::getItem() const {

return item;

}

#endif

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#pragma once

#include<iostream>

#include<string>

using namespace std;

class Watches

{

private:

char\* genderW;

char\* brandW;

double priceW;

public:

// Default Constructor

Watches();

// Constructor with parameters

Watches(char\* aGenderW, char\* aBrandW, double aPriceW);

// Copy Constructor

Watches(const Watches&);

// Destructor

~Watches();

// Mutators

void setGenderW(char\* aGenderW);

void setBrandW(char\* aBrandW);

void setPriceW(double aPriceW);

// Accessors

char\* getGenderW() const;

char\* getBrandW() const;

double getPriceW() const;

friend ostream& operator<<(ostream&, const Watches&);

friend istream& operator>>(istream&, Watches&);

};

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include "Jewelry.h"

Jewelry::Jewelry()

{

char temp[15] = " ";

setGender(temp);

setType(temp);

setWeight(0);

setPrice(0);

setMetal(temp);

}

Jewelry::Jewelry(char\* aGender, char\* aType, int aWeight, double aPrice, char\* aMetal)

{

setGender(aGender);

setType(aType);

setWeight(aWeight);

setPrice(aPrice);

setMetal(aMetal);

}

Jewelry::Jewelry(const Jewelry& obj)

{

setGender(obj.getGender());

setType(obj.getType());

setWeight(obj.getWeight());

setPrice(obj.getPrice());

setMetal(obj.getMetal());

}

Jewelry::~Jewelry()

{

cout << "Destructor de Jewelry invocado\n";

delete[] gender;

delete[] type;

delete[] metal;

}

void Jewelry::setGender(char\* aGender)

{

gender = new char[strlen(aGender) + 1];

strcpy\_s(gender, strlen(aGender) + 1, aGender);

}

void Jewelry::setType(char\* aType)

{

type = new char[strlen(aType) + 1];

strcpy\_s(type, strlen(aType) + 1, aType);

}

void Jewelry::setWeight(int aWeight)

{

weight = aWeight;

}

void Jewelry::setPrice(double aPrice)

{

price = aPrice;

}

void Jewelry::setMetal(char\* aMetal)

{

metal = new char[strlen(aMetal) + 1];

strcpy\_s(metal, strlen(aMetal) + 1, aMetal);

}

char\* Jewelry::getGender() const

{

return gender;

}

char\* Jewelry::getType() const

{

return type;

}

int Jewelry::getWeight() const

{

return weight;

}

double Jewelry::getPrice() const

{

return price;

}

char\* Jewelry::getMetal() const

{

return metal;

}

ostream& operator<<(ostream& strm, const Jewelry& obj)

{

strm << "La joya tiene las siguientes especificaciones:"

<< "\n Gender: " << obj.getGender()

<< "\n Jewelry Type: " << obj.getType()

<< "\n Gold Metal Weight: " << obj.getWeight()

<< "\n Price: $" << obj.getPrice()

<< "\n Metal Type: " << obj.getMetal();

return strm;

}

istream& operator>>(istream& strm, Jewelry& obj)

{

cout << "Escoja el genero para quien sera la joya MUJER, VARON, NINO: ";

strm >> obj.gender;

cout << "Escoja el tipo de joya. Opcionees: ANILLOS, CADENAS, BRAZALETES, ARETES: ";

strm >> obj.type;

cout << "Especifique los Kilates de oro de la joya: ";

strm >> obj.weight;

cout << "Entre el precio: $";

strm >> obj.price;

cout << "Entre el tipo de metal que sera la joya: ";

strm >> obj.metal;

return strm;

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include "Watches.h"

Watches::Watches()

{

char temp[10] = " ";

setGenderW(temp);

setBrandW(temp);

setPriceW(0);

}

Watches::Watches(char\* genderW, char\* brandW, double priceW)

{

setGenderW(genderW);

setBrandW(brandW);

setPriceW(priceW);

}

Watches::Watches(const Watches& obj)

{

setGenderW(obj.getGenderW());

setBrandW(obj.getBrandW());

setPriceW(obj.getPriceW());

}

Watches::~Watches()

{

cout << "Destructor de watches invocado\n";

delete [] genderW;

delete [] brandW;

}

void Watches::setGenderW(char\* aGenderW)

{

genderW = new char[strlen(aGenderW) + 1];

strcpy\_s(genderW, strlen(aGenderW) + 1, aGenderW);

}

void Watches::setBrandW(char\* aBrandW)

{

brandW = new char[strlen(aBrandW) + 1];

strcpy\_s(brandW, strlen(aBrandW) + 1, aBrandW);

}

void Watches::setPriceW(double aPriceW)

{

priceW = aPriceW;

}

char\* Watches::getGenderW() const

{

return genderW;

}

char\* Watches::getBrandW() const

{

return brandW;

}

double Watches::getPriceW() const

{

return priceW;

}

ostream& operator<<(ostream& strm, const Watches& obj)

{

strm << "El reloj que ha escojido tiene estas especificaciones:";

strm << "\n Genero: " << obj.getGenderW();

strm << "\n Brand: " << obj.getBrandW();

strm << "\n Precio: $" << obj.getPriceW();

return strm;

}

istream& operator>>(istream& strm, Watches& obj)

{

cout << "Escoja el genero para quien sera el reloj MUJER, VARON, NINO ";

strm >> obj.genderW;

cout << "Especifique la marca de reloj que busca. Ej. Ferrary, COACH, Casio, Bilova, Citizen, Boos, etc. ";

strm >> obj.brandW;

cout << "Por favor, entre p[recio: $";

strm >> obj.priceW;

return strm;

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include"BoxInterface.h"

#include "JewelryBox.h"

#include "Jewelry.h"

#include "PlainBox.h"

#include "Watches.h"

#include <iostream>

#include <string>

using namespace std;

int main()

{

BoxInterface<Jewelry> \*joyas = new JewelryBox <Jewelry>;

BoxInterface<Watches> \*relojes = new JewelryBox <Watches>;

joyas->setItem() << endl;

relojes->setItem() << endl;

cout << joyas->getItem() << endl;

cout << relojes->getItem() << endl;

delete joyas;

delete relojes;

system("pause");

return 0;

}