Министерство образования Республики Беларусь

Учреждение образования

Брестский государственный технический университет

Кафедра ИИТ

ОТЧЕТ №1

по ООП

Тема: “НАСЛЕДОВАНИЕ И ВИРТУАЛЬНЫЕ ФУНКЦИИ ”

Выполнил:

студент группы АС-36

Марченко А.Г.

\_\_\_\_

Проверил:

Дунец А.П.  
\_\_\_\_

Брест 2015

**Лабораторная работа №8**

**Наследование и виртуальные функции**

**Цель работы:** Получить практические навыки создания иерархии классов и использования статических компонентов класса.

**Постановка задачи:**

Написать программу, в которой создается иерархия классов. Должны быть реализованы классы:игрушка, продукт, товар, молочный продукт. Включить полиморфные объекты в связанный список, используя статические компоненты класса. Показать использование виртуальных функций.

**Вариант 9**

**Иерархия классов в виде графа.**

Продукт

Игрушка

Товар

Молочный продукт

**Код программы:**

**Goods.h**

#ifndef GOODS\_H

#define GOODS\_H

class Goods

{

protected:

double price;

Goods \*next;

public:

static Goods \*head;

static Goods \*end;

virtual void *Show*() = 0;

Goods();

Goods(double);

Goods(double, bool);

~Goods();

void SetPrice(double);

void Add();

static void Print();

};

#endif // GOODS\_H

**Goods.cpp**

#include"goods.h"

#include<iostream>

using namespace std;

Goods\* Goods::head = NULL;

Goods\* Goods::end = NULL;

void Goods::Add()

{

if(end == NULL)

{

head = this;

end = this;

}

if(end == head != NULL)

{

head -> next = this;

end = this;

}

else

{

end -> next = this;

end = this;

}

end->next = NULL;

}

void Goods::SetPrice(double price)

{

this->price = price;

}

void Goods::Print()

{

Goods \*curr = head;

while(curr != NULL)

{

curr->Show();

curr = curr -> next;

}

}

Goods::Goods()

{

price = 0;

}

Goods::Goods(double price)

{

this->price = price;

}

Goods::Goods(double price, bool listItem)

{

this->price = price;

if(listItem)

this->Add();

}

Goods::~Goods()

{

}

**products.h**

#ifndef PRODUCTS\_H

#define PRODUCTS\_H

#include"goods.h"

class Products:public Goods

{

protected:

char \* typeProd;

public:

Products();

Products(char\*, double, bool);

~Products();

void *Show*();

};

#endif // PRODUCTS\_H

**products.cpp**

#include"goods.h"

#include<iostream>

#include"products.h"

#include "cstring"

using namespace std;

Products::Products()

{

typeProd = new char[3];

strcpy(typeProd, "N/A");

}

Products::Products(char \*type, double price, bool listItem)

:Goods(price, listItem)

{

this->typeProd = new char[strlen(type) + 1];

strcpy(this->typeProd, type);

}

Products::~Products()

{

delete[] typeProd;

}

void Products::*Show*()

{

cout << "Products name = " << typeProd

<< " price = " << price << endl;

}

**toys.h**

#ifndef TOYS\_H

#define TOYS\_H

#include"goods.h"

class Toys:public Goods

{

protected:

char \*name;

public:

Toys();

Toys(char\*, double, bool);

~Toys();

void *Show*();

};

#endif // TOYS\_H

**toys.cpp**

#include"goods.h"

#include<iostream>

#include"toys.h"

#include "cstring"

using namespace std;

Toys::Toys()

{

name = new char[3];

strcpy(name, "N/A");

}

Toys::Toys(char \*name, double price, bool listItem)

:Goods(price, listItem)

{

this->name = new char[strlen(name) + 1];

strcpy(this->name, name);

}

Toys::~Toys()

{

delete[] name;

}

void Toys::*Show*()

{

cout << "Toys name = " << name

<< " price = " << price << endl;

}

**dairyProduce.h**

#ifndef DAIRYPRODUCE\_H

#define DAIRYPRODUCE\_H

#include"products.h"

class DairyProduce:public Products

{

protected:

char \*nameOfDP;

public:

DairyProduce();

DairyProduce(char\*, char\*, double, bool);

~DairyProduce();

void *Show*();

};

#endif // DAIRYPRODUCE\_H

**dairyProduce.cpp**

#include"products.h"

#include<iostream>

#include"dairyProduce.h"

#include "cstring"

using namespace std;

DairyProduce::DairyProduce()

{

nameOfDP = new char[3];

strcpy(nameOfDP, "N/A");

}

DairyProduce::DairyProduce(char \*nameOfDP,char\* TypeProd, double price, bool listItem)

:Products(TypeProd, price, listItem)

{

this->nameOfDP = new char[strlen(nameOfDP) + 1];

strcpy(this->nameOfDP , nameOfDP);

}

DairyProduce::~DairyProduce()

{

delete[] nameOfDP;

}

void DairyProduce::*Show*()

{

cout << "DairyProduce name = " << nameOfDP

<< "; Products name = " << typeProd

<< "; price = " << price << endl;

}

**Main.cpp**

#include <QCoreApplication>

#include"goods.h"

#include"products.h"

#include"toys.h"

#include"dairyProduce.h"

#include<iostream>

//using namespace std;

int main(int argc, char \*argv[])

{

QCoreApplication a(argc, argv);

Goods\* obj1 = new Toys("Bear", 300, true);

// obj1->Add();

Products\* prod1 = new Products("orange", 16, false);

prod1->Add();

Products\* dairy = new DairyProduce("Kefir","Dairy", 25, false);

dairy->Add();

Goods::Print();

delete obj1;

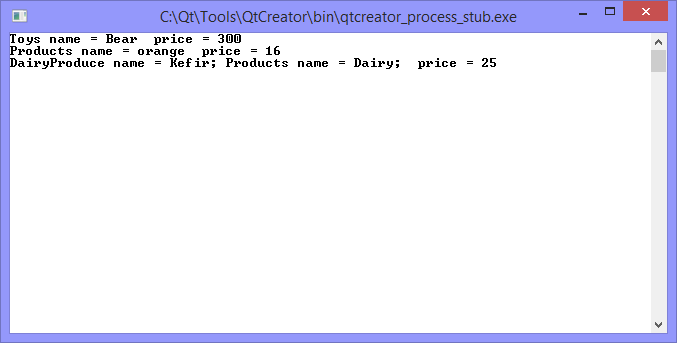
delete prod1;

delete dairy;

return a.exec();

}

**Результат работы программы:**



Если не объявить функцию void *show*() виртуальной, то эта функция будет выводить только значение поля «price», т. е. будет вызываться функция базового класса.

**Результат:**

