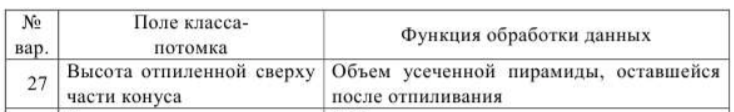
**ЛАБОРАТОРНАЯ РАБОТА № 17. Наследование и полиморфизм. Виртуальные функции**

**Задание:**

Задание 1 (наследование, базовый уровень): для класса, созданного в предыдущем задании (по вариантам табл. 11.1, задание 1 базовый уровень) создать класс-потомок с дополнительным полем, указанным в индивидуальном задании (табл. 11.4, столб 2). Реализовать в классе-потомке методы: - конструктор; 112 - функцию обработки данных, указанную в индивидуальном задании (табл 11.4, столб 3). Создать проект для демонстрации работы: ввод и вывод информации об объектах: классе-родителе и классе-потомке.



Задание 2 (Полиморфизмы методов класса. Базовый уровень):. построить класс 1-го уровня с указанными в индивидуальном задании (табл. 11.7) полями и методами: - конструктор; - функция, которая определяет «качество» объекта – Q по заданной формуле (табл11. 7, столб 2); - вывод информации об объекте. Построить класс 2-го уровня (класс-потомок), который содержит: - дополнительное поле P; - функция, которая определяет «качество» объекта класса 2-го уровня – Qp, которая перекрывает функцию качества класса 1-го уровня (Q ), выполняя вычисление по новой формуле (табл. 11.7, столб 3). Создать проект для демонстрации работы: ввод и вывод информации об объектах классов 1-го и 2-го уровней.



**Листинги программ:**

**Задание 1**

// ---------------------------------------------------------------------------

#include <vcl.h>

#include <math.h>

#include <list>

#pragma hdrstop

#include "Unit1.h"

// ---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

// ---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner) : TForm(Owner) {

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender) {

conus object(StrToFloat(Edit1->Text), StrToFloat(Edit2->Text));

Form1->Base.push\_back(object);

Memo1->Lines->Append(object.Check\_object());

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender) {

Memo1->Lines->Append(Form1->Base.back().Check\_object() + " и объем " + FloatToStr

(Form1->Base.back().calculate\_V()));

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button3Click(TObject \*Sender) {

Memo1->Clear();

Form1->Base.clear();

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button4Click(TObject \*Sender)

{

cut\_conus object(StrToFloat(Edit1->Text), StrToFloat(Edit2->Text),StrToFloat(Edit3->Text));

Memo1->Lines->Append(object.Check\_object() + " и объем " + FloatToStr

(object.v\_cut()));

}

//---------------------------------------------------------------------------

Файл Unit1.h

// ---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

// ---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

// ---------------------------------------------------------------------------

class TForm1 : public TForm {

\_\_published: // IDE-managed Components

TEdit \*Edit1;

TEdit \*Edit2;

TLabel \*Label1;

TLabel \*Label2;

TButton \*Button1;

TMemo \*Memo1;

TButton \*Button2;

TButton \*Button3;

TEdit \*Edit3;

TLabel \*Label3;

TButton \*Button4;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

void \_\_fastcall Button3Click(TObject \*Sender);

void \_\_fastcall Button4Click(TObject \*Sender);

private: // User declarations

public: // User declarations

class conus {

protected:

float r;

float h;

public:

conus(float r, float h) {

this->r = r;

this->h = h;

}

AnsiString Check\_object() {

AnsiString str = "Конус имеет радиус " + FloatToStr(r)

+ " и высоту " + FloatToStr(h);

return str;

}

// функция вычисления объема конуса

double calculate\_V() {

return(1.0 / 3.0) \* 3.14 \* pow(r, 2.0) \* h;

}

};

class cut\_conus : public conus {

protected:

float h\_cut;

public:

cut\_conus(float r, float h, float h\_cut) : conus(r, h) {

this->h\_cut = h\_cut;

}

float v\_cut() {

float r2 = (h\_cut \* r) / h;

return(1.0 / 3.0) \* 3.14 \* h\_cut \* (pow(r, 2.0) + r \* r2 + pow(r,2.0));

}

AnsiString Check\_object() {

AnsiString str = "Учеченный конус имеет радиус " + FloatToStr(r)

+ " и высоту " + FloatToStr(h\_cut);

return str;

}

};

std::list<conus>Base;

\_\_fastcall TForm1(TComponent\* Owner);

};

// ---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

// ---------------------------------------------------------------------------

#endif

**Задание 2**

// ---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

// ---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

// ---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner) : TForm(Owner) {

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender) {

firm object(Edit1->Text, StrToFloat(Edit2->Text), StrToFloat(Edit3->Text));

Memo1->Lines->Append(object.output() + ". " + object.calc());

}

// ---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button2Click(TObject \*Sender) {

second\_firm object(Edit1->Text, StrToFloat(Edit2->Text),

StrToFloat(Edit3->Text), StrToFloat(Edit4->Text));

Memo1->Lines->Append(object.output() + ". " + object.calc());

}

// ---------------------------------------------------------------------------

Файл Unit1.h

// ---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

// ---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <math.h>

// ---------------------------------------------------------------------------

class TForm1 : public TForm {

\_\_published: // IDE-managed Components

TEdit \*Edit1;

TEdit \*Edit2;

TMemo \*Memo1;

TButton \*Button1;

TButton \*Button2;

TEdit \*Edit3;

TEdit \*Edit4;

TLabel \*Label1;

TLabel \*Label2;

TLabel \*Label3;

TLabel \*Label4;

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

private: // User declarations

public: // User declarations

class firm {

protected:

AnsiString Name;

float Income;

float Rate;

public:

firm(AnsiString Name, float Income, float Rate) {

this->Name = Name;

this->Income = Income;

this->Rate = Rate;

}

virtual AnsiString calc() {

float temp = Income \* Rate;

return "Оценка по первому классу" + FloatToStr(temp);

}

virtual AnsiString output() {

return "Доход фирмы " + Name + " : " + Income + " Рейтинг " + Rate;

}

};

class second\_firm : public firm {

protected:

float Investment;

public:

second\_firm(AnsiString Name, float Income, float Rate,

float Investment) : firm(Name, Income, Rate) {

this->Investment = Investment;

}

virtual AnsiString calc() {

float temp = pow(Investment, 3.0) + (Income \* Rate);

return "Оценка по второму классу" + FloatToStr(temp);

}

};

\_\_fastcall TForm1(TComponent\* Owner);

};

// ---------------------------------------------------------------------------

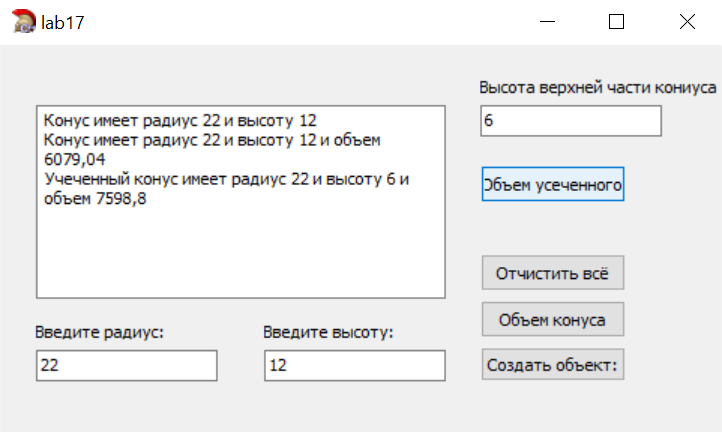
extern PACKAGE TForm1 \*Form1;

// ---------------------------------------------------------------------------

#endif

**Результаты работы программ:**

**Задание 1**



**Задание 2**

