**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ им. И. РАЗЗАКОВА**

**Кафедра «Программное обеспечение компьютерных систем»**

**Отчет**

**ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ С++**

**Лабораторная работа №7**

**Выполнила: Эмилбекова Элиза ПИ-2-21**

**Проверил: Мусабаев Э.Б.**

**Бишкек 2024**

**Тема: Виртуальные методы**

**Задания:**

1. Создать базовый класс Array с полями: массив типа unsigned и поле для хранения количества элементов у текущего объекта массива. Максимально возможный размер массива задается статической константой. Реализовать конструктор инициализации, задающий количество элементов и начальное значение (по умолчанию 0). Реализовать в классе Array виртуальную функцию поэлементного сложения массивов. Реализовать два класса, переопределив виртуальную функцию сложения. Вызывающая программа должна продемонстрировать все варианты вызова виртуальных функций.

**Код:**

#pragma once

#include <iostream>

using namespace std;

class Array {

protected:

static const int SIZE = 10;

unsigned int arr[SIZE]{};

unsigned int elementsCount = 0;

public:

int getSize() const {

return elementsCount;

}

void addElement(unsigned int value) {

arr[elementsCount++] = value;

}

void print() const {

for (int i = 0; i < elementsCount; i++)

cout << arr[i] << " ";

cout << endl;

}

unsigned int operator[] (unsigned int index) const {

return arr[index];

}

virtual Array sum(const Array& right) {

return Array();

}

};

class ArratFirst : public Array {

public:

Array sum(const Array& right) override {

Array result;

for (int i = 0; i < this->elementsCount; i++) {

result.addElement(arr[i] + right[i]);

}

return result;

}

};

class Task1 {

public:

static void run() {

Array arr1;

arr1.addElement(5);

arr1.addElement(2);

arr1.addElement(10);

ArratFirst arr2;

arr2.addElement(2);

arr2.addElement(7);

arr2.addElement(20);

Array arrayResult = arr2.sum(arr1);

arrayResult.print();

}

};

Код представляет классы для работы с массивами, где базовый класс **Array** имеет виртуальный метод **sum**, а производный класс **ArratFirst** переопределяет этот метод для суммирования элементов двух массивов. Класс **Task1** демонстрирует использование этих классов для сложения и вывода результатов на экран.

1. ***\*Создать абстрактный базовый класс Figure с виртуальными методами вычисления площади и периметра. Создать производные класса: Rectangle (прямоугольник), Circle (круг), Trapezium (трапеция) со своими функциями площади и периметра. Самостоятельно определить, какие поля необходимы, какие из них можно задать в базовом классе, а какие в производных. Площадь трапеции: *S = (a + b) x h/2.***

#pragma once

#include <iostream>

using namespace std;

class Figure {

public:

virtual double calculateArea() const = 0;

virtual double calculatePerimeter() const = 0;

};

class Rectangle : public Figure {

private:

double width;

double height;

public:

Rectangle(double w, double h) : width(w), height(h) {}

double calculateArea() const override {

return width \* height;

}

double calculatePerimeter() const override {

return 2 \* (width + height);

}

};

class Circle : public Figure {

private:

double radius;

const double PI = 3.14159;

public:

Circle(double r) : radius(r) {}

double calculateArea() const override {

return PI \* radius \* radius;

}

double calculatePerimeter() const override {

return 2 \* PI \* radius;

}

};

class Trapezium : public Figure {

private:

double base1;

double base2;

double height;

public:

Trapezium(double b1, double b2, double h) : base1(b1), base2(b2), height(h) {}

double calculateArea() const override {

return (base1 + base2) \* height / 2;

}

double calculatePerimeter() const override {

return base1 + base2 + 2 \* sqrt((base1 - base2) \* (base1 - base2) / 4 + height \* height);

}

};

class Task2 {

public:

static void run() {

Rectangle rectangle(5, 10);

Circle circle(3);

Trapezium trapezium(3, 7, 4);

cout << "Rectangle: Area = " << rectangle.calculateArea() << ", Perimeter = " << rectangle.calculatePerimeter() << endl;

cout << "Circle: Area = " << circle.calculateArea() << ", Perimeter = " << circle.calculatePerimeter() << endl;

cout << "Trapezium: Area = " << trapezium.calculateArea() << ", Perimeter = " << trapezium.calculatePerimeter() << endl;

}

};

Код представляет классы для вычисления площади и периметра геометрических фигур: прямоугольника, круга и трапеции. Класс Task2 демонстрирует использование этих классов для вычисления и вывода площади и периметра каждой фигуры на экран.

1. Создать абстрактный базовый класс Currency (валюта) для работы с денежными суммами. Определить виртуальные функции перевода в рубли и вывода на экран. Реализовать производные классы Dollar (доллар) и Euro (евро) со своими функциями перевода и вывода на экран.

#pragma once

#include <iostream>

using namespace std;

class Currency {

public:

virtual double convertToRubles(double amount) const = 0;

virtual void display() const = 0;

};

class Dollar : public Currency {

public:

double convertToRubles(double amount) const override {

return amount \* 70;

}

void display() const override {

cout << "Amount in Dollars" << endl;

}

};

class Euro : public Currency {

public:

double convertToRubles(double amount) const override {

return amount \* 80;

}

void display() const override {

cout << "Amount in Euros" << endl;

}

};

class Task3 {

public:

static void run() {

Dollar dollar;

Euro euro;

double amountInDollars = 100;

double amountInEuros = 150;

cout << "Amount in Rubles (Dollars): " << dollar.convertToRubles(amountInDollars) << endl;

dollar.display();

cout << "Amount in Rubles (Euros): " << euro.convertToRubles(amountInEuros) << endl;

euro.display();

}

};

Код представляет классы для работы с денежными суммами в различных валютах: Dollar (доллар) и Euro (евро), а также класс Task3 для демонстрации их функциональности. Классы наследуются от абстрактного базового класса Currency, который определяет виртуальные методы для конвертации в рубли и отображения суммы на экране. Метод Task3::run() создает объекты Dollar и Euro, выполняет конвертацию сумм в рубли и выводит результаты на экран.