Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

**Лабораторная работа №2**

по курсу “Объектно ориентированное программирование”

тема: Наследование

Выполнил студент группы I2302:

Ciobanu Stanislav.,

Проверил преподаватель:

G.Latul

Кишинэу, 2023

**Условия работы**

Вариант 1 : Создайте иерархию классов Сторона – Прямоугольник – Тумбочка с одной полкой.

1. Класс Прямоугольник должен содержать метод для вычисления площади прямоугольника.
2. Класс Тумбочка должен содержать метод для вычисления вместительности
3. Последние два класса в иерархии должны иметь конструкторы.
4. Создайте метод MAIN, в котором создаётся 2 прямоугольника (т.е. 2 объекта класса Прямоугольник)
5. Определить какой из прямоугольников больше и сколько раз меньший прямоугольник входит в большой.
6. Необходимо также показывать все характеристики создаваемых объектов.

**Задние 0**

Создаём классы Side, Rectangle, OneShelfTable в отдельных header файлах.

// Side.h

class Side

{

protected:

int \_length;

public:

int GetLength();

virtual void PrintCharacteristics();

};

// Rectangle.h

class Rectangle : public Side

{

protected:

int \_width;

public:

Rectangle(int length, int width);

Rectangle() :Rectangle(10, 10) {}

int GetWidth();

int GetArea();

virtual void PrintCharacteristics();

void CompareWithRectangle(Rectangle rect);

};

// OneShelfTable.h

class OneShelfTable : public Rectangle

{

private:

int \_height;

int GetCapacity();

public:

OneShelfTable(int length, int width, int height);

void PrintCharacteristics();

};

Далее будем имплементировать все методы в соответствующих .cpp файлах.

**Задние 1**

Создаём внутри класса Rectangle метод GetArea и реализуем его в Rectangle.cpp

// Rectangle.cpp

int Rectangle::GetArea()

{

return \_width \* \_length;

}

**Задние 2**

Создаём внутри класса OneShelfTable метод GetCapacity и реализуем его в OneShelfTable.cpp

// OneShelfTable.cpp

int OneShelfTable::GetCapacity()

{

return GetArea() \* \_height;

}

**Задние 3**

Создаём конструкторы классам OneShelfTable и Rectangle. Реализуем их в соответствующих .cpp файлах.

// OneShelfTable.cpp

OneShelfTable::OneShelfTable(int length, int width, int height)

{

\_length = length;

\_width = width;

\_height = height;

}

// Rectangle.cpp

Rectangle::Rectangle(int length, int width)

{

\_length = length;

\_width = width;

}

**Задние 4**

Внутри метода main создаём 2 экземпляра класса Rectangle

// OOPlab3.cpp

#include <iostream>

#include "Side.h"

#include "Rectangle.h"

#include "OneShelfTable.h"

using namespace std;

. . .

int main()

{

Rectangle rect1 = \*new Rectangle(5, 12);

Rectangle rect2 = \*new Rectangle(10, 12);

. . .

}

. . .

**Задние 5**

Создаём функцию, которая будет решать, какой из прямоугольников больше и сколько раз меньший прямоугольник входит в большой – CompareArea.

// OOPlab3.cpp

. . .

void CompareArea(Rectangle rect1, Rectangle rect2);

int main()

{

. . .

cout << endl;

CompareArea(rect1, rect2);

}

void CompareArea(Rectangle rect1, Rectangle rect2)

{

int area1 = rect1.GetArea();

int area2 = rect2.GetArea();

if (area1 > rect2.GetArea())

{

. . .

}

else if (area1 < area2)

{

. . .

}

else

{

. . .

}

}

**Задние 6**

Создаём метод ShowCharacteristics в классе Side. Наследуем этот метод классу Rectangle и от него классу OneShelfTable. Делаем этот метод виртуальным внутри классов Side и Rectangle. Реализуем их в соответствующих .cpp файлах.

// Side.cpp

void Side::PrintCharacteristics()

{

cout << " Length : " << \_length << endl;

}

// Rectangle.cpp

void Rectangle::PrintCharacteristics()

{

cout << " Length : " << \_length << " Width : " << \_width << endl;

}

// OneShelfTable.cpp

void OneShelfTable::PrintCharacteristics()

{

cout << " Length : " << \_length << " Width : " << \_width << " Heigtht : " << \_height << endl;

}

// OOPlab3.cpp

int main()

{

. . .

rect1.PrintCharacteristics();

rect2.PrintCharacteristics();

. . .

}

**Используемые библиотеки**

<iostream> // Console

"Side.h" // Headers

"Rectangle.h"

"OneShelfTable.h"