Пензенский государственный технологический университет

Факультет автоматизированных информационных технологий

Кафедра «Прикладная информатика»

ЕН.Ф.02

**ПОЛИМОРФИЗМ**

Отчет о лабораторной работе № 6

по дисциплине «Информатика и программирование»

Выполнили: ст-ты гр.17СН1с

Тюняев А.В., Богданов М.О.

Проверил: ст. преп. каф. ПИ

Юранов В.С.

2018

**1 Формулировка задачи**

1. В приложение из 3 лабораторной работы необходимо добавить класс Walls для автоматического создания стен игрового поля.
2. В основной программе создать объект класса Walls.
3. В конструкторе класса указать размеры стены равными размеру экрана консоли.

Пример работающей программы:

**2 Требования к программе**

Программа должна вывести границы экрана.

**3 Описание программы**

**3.1 Общие сведения**

Программа разработана в среде Visual Studio 2017

**3.2 Функциональное назначение**

Программа предназначена для вывода границ экрана

**3.3 Описание логической структуры**

*Класс Point:*

В классе объявляются три поля, значение оси x, значение оси y, символ, который будет выводится.

Перегрузка конструктора Point.

Конструктор Point с тремя параметрами.

Конструктор Point с одним параметром.

Метод Draw служит для вывода символов на экран.

Метод IsHit используется для проверки точек на пересечения.

*Класс Figure:*

Создается список pList для записи координат точек

Метод Draw служит для вывода символов на экран используя список pList.

Метод IsHit используется для проверки фигур на пересечение.

Метод IsHit используется для проверки точек на пересечения.

*Класс VerticalLine:*

Этот класс является наследником класса Figure.

Метод VerticalLine принимает значения смещения по оси y, координату по оси x и символ.

Создается экземпляр класса List<Point> (списка).

В цикл передаются значения начальной и конечной точки по оси y.

В цикле создается экземпляр класса Point и в список добавляются координаты точек по осям x и y, и необходимый символ.

*Класс HorizontalLine:*

Этот класс является наследником класса Figure.

Метод HorizontalLine принимает значения смещения по оси x, координату по оси y и символ.

Создается экземпляр класса List<Point> (списка).

В цикл передаются значения начальной и конечной точки по оси x.

В цикле создается экземпляр класса Point и в список добавляются координаты точек по осям x и y, и необходимый символ.

*Класс Program:*

Метод Main (основной метод) в нем создается экземпляр класса Walls, и передаются параметры. Вызывается метод Draw класса Walls, для вывода границ экрана.

*Класс Walls:*

Создается список wallList.

В методе Walls

Создаются экземпляры классов HorizontalLine, VerticalLine и передаются параметры.

Происходит добавление прямых в список wallList.

С помощью метода Draw происходит вывод фигур на экран.

**3 Описание применения**

После запуска программы выводятся 4 линии, которые очерчивают границы экрана.

**Выводы**

Познакомились с понятием динамического полиморфизма.

**Контрольные вопросы**

1. **Что такое полиморфизм?**

Полиморфи́зм — возможность объектов с одинаковой спецификацией (именем) иметь различную реализацию.

1. **Какие вида полиморфизма существуют?**

* Ad-hoc полифорфизм (перегрузка функций, приведение типа) — поддерживается, можно создать несколько методов с одним именем и разными аргументами, примитивные типы могут неявно приводиться.
* Параметрический полиморфизм (обобщённое программирование) — поддерживается, можно создать полиморфные (обобщённые) типы.
* Полиморфизм включения (наследование) — поддерживается, есть интерфейсы, наследование классов, одиночная и множественная диспетчеризация. Примитивные типы неполиморфны.

1. **Что такое абстрактный класс?**

Абстрактный класс — базовый класс, который не предполагает создания экземпляров. Абстрактные классы реализуют на практике один из принципов ООП — полиморфизм. Абстрактный класс может содержать абстрактные методы и свойства. Абстрактный метод не реализуется для класса, в котором описан, однако должен быть реализован для его неабстрактных потомков. Абстрактные классы представляют собой наиболее общие абстракции, то есть имеющие наибольший объём и наименьшее содержание.

1. **Что такое интерфейс?**

С помощью интерфейса можно сделать общий метод для двух классов (оба класса будут наследниками одного интерфейса в котором будет прописан метод)

Еще одно применение интерфейса — быстрое изготовление заглушек для функций.

Например, вы в команде строите большой проект и ваш класс зависит от класса, который пишет коллега. Но еще не написал.

Главное отличие класса от интерфейса — в том, что класс состоит из интерфейса и реализации.

1. **Для чего нужно переопределение функций?**

Перегрузка методов относится к одному из способов реализации полиморфизма в C#.

Сокращение кода, удобство.

1. **С помощью чего реализуется полиморфизм в программе?**

С помощью перегрузки методов.

ТЕКСТ ПРОГРАММЫ

Приложение А

(обязательное)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab6

{

class Point

{

int x;

int y;

char sym;

public Point(int x, int y, char sym)

{

this.x = x;

this.y = y;

this.sym = sym;

}

public Point(Point p)

{

x = p.x;

y = p.y;

sym = p.sym;

}

public void Draw()

{

Console.SetCursorPosition(x, y);

Console.Write(sym);

}

public bool IsHit(Point p)

{

return p.x == this.x && p.y == this.y;

}

}

class Figure

{

protected List<Point> pList;

public void Draw()

{

foreach (Point p in pList)

{

p.Draw();

}

}

public bool IsHit(Figure figure)

{

foreach (var p in pList)

{

if (figure.IsHit(p))

return true;

}

return false;

}

private bool IsHit(Point point)

{

foreach (var p in pList)

{

if (p.IsHit(point))

return true;

}

return false;

}

}

class VerticalLine : Figure

{

public VerticalLine(int yUp, int yDown, int x, char sym)

{

pList = new List<Point>();

for (int y = yUp; y <= yDown; y++)

{

Point p = new Point(x, y, sym);

pList.Add(p);

}

}

}

class HorizontalLine : Figure

{

public HorizontalLine(int xLeft, int xRight, int y, char sym)

{

pList = new List<Point>();

for (int x = xLeft; x <= xRight; x++)

{

Point p = new Point(x, y, sym);

pList.Add(p);

}

}

}

class Program

{

static void Main(string[] args)

{

Walls wl = new Walls(80,30);

wl.Draw();

Console.ReadKey();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab6

{

class Walls

{

List<Figure> wallList;

public Walls(int mapWidth, int mapHeight)

{

wallList = new List<Figure>();

HorizontalLine upLine = new HorizontalLine(0, mapWidth - 2, 0, '+');

HorizontalLine downLine = new HorizontalLine(0, mapWidth - 2, mapHeight - 1, '+');

VerticalLine leftLine = new VerticalLine(0, mapHeight - 1, 0, '+');

VerticalLine rightLine = new VerticalLine(0, mapHeight - 1, mapWidth - 2, '+');

wallList.Add(upLine);

wallList.Add(downLine);

wallList.Add(leftLine);

wallList.Add(rightLine);

}

public bool IsHit(Figure figure)

{

foreach (var wall in wallList)

{

if (wall.IsHit(figure))

{

return true;

}

}

return false;

}

public void Draw()

{

foreach (var wall in wallList)

{

wall.Draw();

}

}

}

}

РЕЗУЛЬТАТЫ ИСПЫТАНИЙ

Приложение Б

(обязательное)

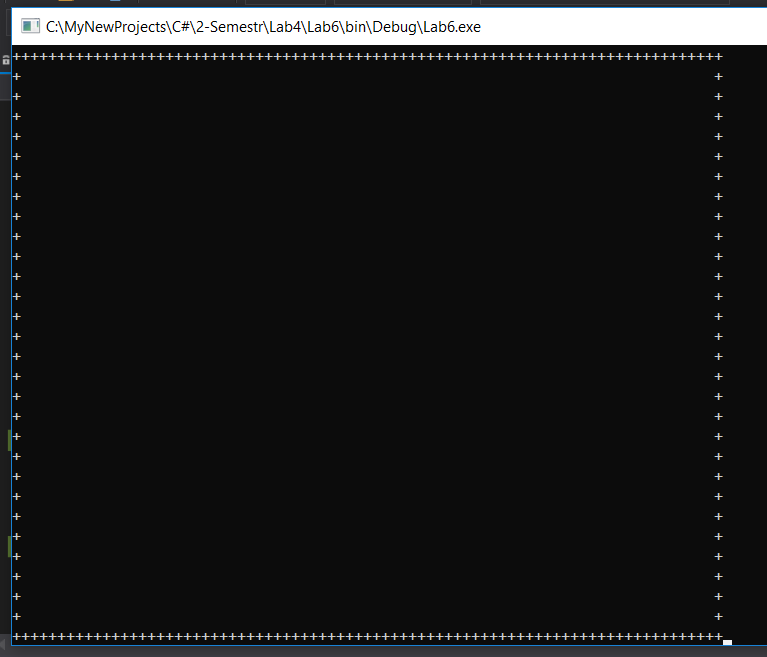


Рисунок 1 – Результат выполнения программы