Пензенский государственный технологический университет

Факультет автоматизированных информационных технологий

Кафедра «Прикладная информатика»

ЕН.Ф.02

**ПЕРЕГРУЗКА КОНСТРУКТОРОВ И МЕТОДОВ**

Отчет о лабораторной работе № 5

по дисциплине «Информатика и программирование»

Выполнил: ст-т гр.17СН1с

Тюняев А.В., Богданов М.О.

Проверил: ст. преп. каф. ПИ

Юранов В.С.

2018

**1 Формулировка задачи**

1. В приложение из 2 лабораторной работы необходимо добавить функции для определения пересечений между точками и фигурами.
2. В основной программе создать объект класса Point через конструктор с числовыми параметрами.
3. Создать второй объект класса Point через конструктор с параметром типа Point – для того чтобы получить две одинаковые точки.
4. Проверить работу функции IsHit(Point p) на одной из созданных точек(в качестве аргумента указать вторую созданную точку).
5. Убедиться, что функция IsHit(Point p) возвращает true. (вывести это на экран)
6. Изменить координату одной из точек и убедиться, что функция IsHit(Point p) возвращает false. (вывести это на экран)
7. Создать две линии – вертикальную и горизонтальную так, чтобы они пересекались.
8. Проверить работу функции IsHit( Figure figure ) на одной из линий, при этом в качестве аргумента указать экземпляр второй линии. (вывести это на экран)

**2 Требования к программе**

Программа должна вывести на экран две точки и две прямые и проверить пресекаются ли между собой прямые и точки.

**3 Описание программы**

**3.1 Общие сведения**

Программа разработана в среде Visual Studio 2017

**3.2 Функциональное назначение**

Программа предназначена для вывода символов на экран и проверки на пересечение.

**3.3 Описание логической структуры**

*Класс Point:*

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

Конструктор Point ….

Конструктор Point…

Метод Draw служит для вывода символов на экран.

Метод IsHit ….

*Класс Figure:*

Создается список pList для записи координат точек

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

Метод IsHit …

*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 экземпляры классов VerticalLine и HorizontalLine, передаются данные в эти классы, вызывается метод Draw, для вывода рисунка.

**3 Описание применения**

После запуска программы выводятся две линии и две точки, программа проверяет пересекаются ли между собой линии и пересекаются ли точки, и выводит результат.

**Выводы**

Познакомились с созданием нескольких конструкторовдля класса и полиморфизмом методов.#.

**Контрольные вопросы**

1. Что такое конструктор?
2. Зачем нужен конструктор?
3. Что такое наследование?
4. Что такое методы и поля класса?
5. Что такое модификаторы доступа и для чего они нужны?
6. Что такое статический полиморфизм?
7. Что такое переопределение функций?

ТЕКСТ ПРОГРАММЫ

Приложение А

(обязательное)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab5

{

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 Checks

{

public void Check(Point x, Point y)

{

if ((x.IsHit(y)) == true)

Console.WriteLine("Точки совпадают");

else

Console.WriteLine("Точки не совпадают");

}

public void Check(HorizontalLine x, VerticalLine y)

{

if (x.IsHit(y) == true)

Console.WriteLine("Линии пересекаются");

else

Console.WriteLine("Линии не пересекаются");

}

}

class Program

{

static void Main(string[] args)

{

Checks check = new Checks();

Point p = new Point(62, 2, '\*');

p.Draw();

Point p2 = new Point(p);

p2.Draw();

Console.SetCursorPosition(55, 7);

check.Check(p, p2);

Point p3 = new Point(80, 2, '\*');

p3.Draw();

Point p4 = new Point(85, 4, '\*');

p4.Draw();

Console.SetCursorPosition(75, 7);

check.Check(p3, p4);

HorizontalLine h1 = new HorizontalLine(1, 10, 2, '\*');

h1.Draw();

VerticalLine v1 = new VerticalLine(1, 5, 12, '\*');

v1.Draw();

Console.SetCursorPosition(1, 7);

check.Check(h1, v1);

HorizontalLine h2 = new HorizontalLine(35, 45, 2, '\*');

h2.Draw();

VerticalLine v2 = new VerticalLine(1, 5, 40, '\*');

v2.Draw();

Console.SetCursorPosition(30, 7);

check.Check(h2, v2);

Console.ReadKey();

}

}

}

РЕЗУЛЬТАТЫ ИСПЫТАНИЙ

Приложение Б

(обязательное)

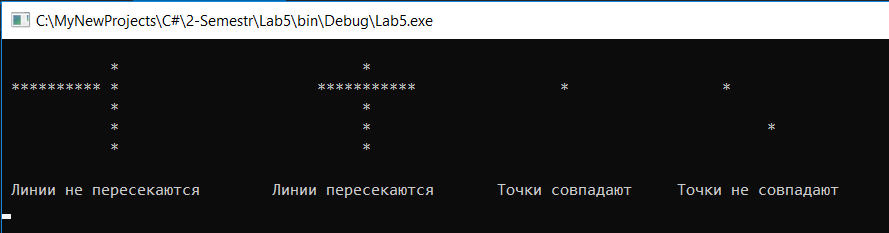


Рисунок 1 – Результат выполнения программы