|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет экономики, менеджмента и бизнес-информатики* |
|  |
| Меньшиков Олег Владимирович  *Лабораторная 9*  по направлению подготовки *09.03.04 Программная инженерия*  образовательная программа «Программная инженерия»   |  |  | | --- | --- | |  | Доцент к[афедры информационных технологий в бизнесе](https://perm.hse.ru/bi/infobis/), кандидат технических наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  О.Л. Викентьева | |

# **Анализ**

## Часть 1.

1. Реализовать (в отдельном файле) определение нового класса (закрытые атрибуты, свойства, конструкторы, инициализация и вывод атрибутов).
2. Для демонстрации работы с объектами написать главную функцию, в которой создаются объекты класса и выводится информация, которая содержится в атрибутах.
3. Написать функцию, реализующую указанное в варианте действие. Рассмотреть два варианта:

1) статическую функцию;

2) метод класса.

В основной функции продемонстрировать работу функции.

1. Используя статическую компоненту класса подсчитать количество созданных в программе объектов.

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | Triangle | double a, double b, double c (длины сторон треугольника) | Определить существует ли треугольник с такими длинами сторон. Результат должен быть типа bool. |

### Часть 2.

1. Добавить к реализованному классу указанные в варианте перегруженные операции.
2. Написать демонстрационную программу, в которой создаются объекты пользовательских классов и выполняются указанные операции.

|  |  |  |
| --- | --- | --- |
| 10 | Triangle | Унарные операции:  – вычисляется площадь треугольника, результатом должно быть вещественное число.  Операции приведения типа:  double (неявная) – результатом является периметр треугольника;  bool (явная) – результатом является true, если треугольник с такими длинами сторон существует и false в противном случае.  Бинарные операции:  <- сравниваются площади треугольников;  > - сравниваются площади треугольников. |

## Часть 3

1. Реализовать класс (в отдельном файле), полем которого является одномерный массив из элементов заданного в варианте типа. В классе реализовать:

* конструктор без параметров,
* конструктор с параметрами, заполняющий элементы случайными значениями,
* конструктор с параметрами, позволяющий заполнить массив элементами, заданными пользователем с клавиатуры,
* индексатор (для доступа к элементам массива),
* метод для просмотра элементов массива.

1. Написать демонстрационную программу, позволяющую создать массив разными способами и распечатать элементы массива. Подсчитать количество созданных объектов.
2. Выполнить указанное в варианте задание (если необходимо, перегрузить нужные для выполнения задачи операции или функции).

# **Листинг**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab9

{

public class Triangle

{

//переменные со свойствами

private static int count = 0;

public static int Count

{

get { return count; }

}

double a;

public double A

{

set

{

if (value > 0) a = value;

else Console.WriteLine("ДЛИНА СТОРОНЫ ДОЛЖНА БЫТЬ БОЛЬШЕ НУЛЯ!");

}

get { return a; }

}

double b;

public double B

{

set

{

if (value > 0) b = value;

else Console.WriteLine("ДЛИНА СТОРОНЫ ДОЛЖНА БЫТЬ БОЛЬШЕ НУЛЯ!");

}

get { return b; }

}

double c;

public double C

{

set

{

if (value > 0) c = value;

else Console.WriteLine("ДЛИНА СТОРОНЫ ДОЛЖНА БЫТЬ БОЛЬШЕ НУЛЯ!");

}

get { return c; }

}

static Random rnd = new Random();

//конструкторы

public Triangle(double a, double b, double c)

{

this.a = a;

this.b = b;

this.c = c;

if (ExistFunc())

count++;

}

public Triangle()

{

this.a = Math.Round(rnd.NextDouble() \* rnd.NextDouble() \* 10 + 1, 1);

this.b = Math.Round(rnd.NextDouble() \* rnd.NextDouble() \* 10 + 1, 1);

this.c = Math.Round(rnd.NextDouble() \* rnd.NextDouble() \* 10 + 1, 1);

if(ExistFunc())

count++;

}

//функции

public void Show()

{

Console.WriteLine("===============================================================================================");

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(" Информация о данном треугольнике:");

if (ExistFunc())

{

Console.WriteLine($"a = {this.a}; b = {this.b}; c = {this.c}\nТреугольник существует!");

}

else

Console.WriteLine($"a = {this.a}; b = {this.b}; c = {this.c}\nТреугольник не существует, т.к. одна сторона не может быть больше оставшихся или равняться им!");

Console.WriteLine("===============================================================================================\n");

Console.ResetColor();

}

public static bool ExistStat(Triangle Tr)

{

if ((Tr.a >= Tr.b + Tr.c) || (Tr.b >= Tr.a + Tr.c) || (Tr.c >= Tr.a + Tr.b))

return false;

return true;

}

public bool ExistFunc()

{

if ((a >= b + c) || (b >= a + c) || (c >= a + b))

return false;

return true;

}

//унарные операции

public static double operator -(Triangle T)

{

if (!ExistStat(T)) return 0;

double p = (T.A + T.B + T.C) / 2;

return Math.Round(Math.Sqrt(p \* (p - T.A) \* (p - T.B) \* (p - T.C)), 1);

}

//бинарные операции

public static bool operator >(Triangle T1, Triangle T2)

{

if (-T1 > -T2) return true;

return false;

}

public static bool operator <(Triangle T1, Triangle T2)

{

if (-T1 < -T2) return true;

return false;

}

//операции приведения типа

public static implicit operator double(Triangle T)

{

double P = T.A + T.B + T.C;

return P;

}

public static explicit operator bool(Triangle T)

{

if (ExistStat(T))

return true;

return false;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace лаба9

{

class Program

{

static double ParsingDouble()

{

bool ok;

double n;

do

{

string buf = Console.ReadLine();

ok = double.TryParse(buf, out n);

if ((!ok) || (n <= 0)) Console.WriteLine("Значение введено неверно! Введите заново: ");

}

while ((!ok) || (n <= 0));

return n;

}

static int ParsingInt(string buf)

{

bool ok;

int n;

do

{

ok = int.TryParse(buf, out n);

if ((!ok) || (n <= 0))

{

Console.WriteLine("Значение введено неверно! Введите заново: ");

buf = Console.ReadLine();

}

}

while ((!ok) || (n <= 0));

return n;

}

static public Triangle CreateKeyBoard()

{

Console.Write("Введите длину 1 стороны: ");

double storona1 = ParsingDouble();

Console.Write("Введите длину 2 стороны: ");

double storona2 = ParsingDouble();

Console.Write("Введите длину 3 стороны: ");

double storona3 = ParsingDouble();

var Tr = new Triangle(storona1, storona2, storona3);

Tr.Show();

return Tr;

}

static public Triangle CreateRandom()

{

var Tr = new Triangle();

Tr.Show();

return Tr;

}

static Triangle CreateTriangle()

{

Console.WriteLine("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<СОЗДАНИЕ НОВОГО ТРЕУГОЛЬНИКА>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>");

while (true)

{

Console.Write("1. Через клавитуру.\n2. Датчик Случайных Чисел.\nВаш выбор: ");

int i = ParsingInt(Console.ReadLine());

Console.WriteLine("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>");

switch (i)

{

case 1:

return CreateKeyBoard();

case 2:

return CreateRandom();

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

}

static void ExistingTriangle()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

var Tr1 = new Triangle(5, 5, 5);

Console.WriteLine("1. Создадим равносторонний треугольник со сторонами, равными 5:");

Console.ResetColor();

Tr1.Show();

}

static void NonExistingTriangle()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

var Tr2 = new Triangle(1, 1, 5);

Console.WriteLine("2. Создадим несуществующий треугольник со сторонами 1, 1, 5: ");

Console.ResetColor();

Tr2.Show();

}

static void ZeroAndNegativeSide()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

var Tr2 = new Triangle(4, 5, 12);

Console.WriteLine("5. Присвоим отрицательные и нулевые значения сторонам треугольника:\n");

Console.Write("<>Присвоим стороне значение -5, программа выведет: ");

Tr2.A = -5;

Console.Write("<>Присвоим стороне значение 0, программа выведет: ");

Tr2.A = 0;

Console.WriteLine("\n");

Console.ResetColor();

}

static void StaticAndDynamicFunc()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

var Tr4 = new Triangle(15, 17, 16);

Console.WriteLine("3. Создадим треугольник со сторонами 15, 17, 16: \n");

Console.WriteLine("Проверим работу динамического и статического метода класса Triangle");

Console.WriteLine(" <>Статический метод даст значение: " + Triangle.ExistStat(Tr4));

Console.WriteLine(" <>Динамический метод даст значение: " + Tr4.ExistFunc());

Console.WriteLine("Видно, что внешне оба метода дают одинаковый результат, разница имеется лишь в ситаксисе.\n");

Console.ResetColor();

}

static void MenuPart1()

{

Console.WriteLine("------------------------------МЕНЮ 1-----------------------------------------------------------");

Console.Write("1. Существующий треугольник.\n2. Несуществующий треугольник.\n" +

"3. Статический и динамический методы.\n" +

"4. Создать свой треугольник.\n5. Треугольник с отрицательной и нулевой стороной.\n" +

"6. Количество корректно созданных треугольников.\n7. Перейти к ЧАСТЬ №2.\n" +

"----------------------------------------------------------------------------------------------- " +

"\nВаш выбор: ");

}

static void MenuPart2()

{

Console.WriteLine("\n------------------------------МЕНЮ 2-----------------------------------------------------------");

Console.Write("1. Унарная операция '-'(находит площадь).\n2. Бинарная операция '>'(сравнение площадей).\n" +

"3. Бинарная операция '<'(сравнение площадей).\n" +

"4. Неявное приведение типа(находит периметр).\n5. Явное приведение типа(существование треугольника).\n" +

"6. Количество корректно созданных треугольников.\n7. Перейти к ЧАСТЬ №3.\n" +

"-----------------------------------------------------------------------------------------------" +

"\nВаш выбор: ");

}

//унарные

static void UnarnOpDemonstr()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.Write("1. Унарная операция '-':\n<>Создадим существующий треугольник со сторонами 6, 7, 8." +

"\n\tПрименим '-' " +

"и программа выведет площадь: ");

var Tr = new Triangle(6, 7, 8);

double S = -Tr;

Console.WriteLine(S);

var Tr2 = new Triangle(4, 4, 90);

Console.Write("<>Создадим несуществующий треугольник со сторонами 4, 4, 90. \n\tПрименим '-' " +

"и программа выведет: ");

UnarnOp(Tr2);

Console.ResetColor();

Console.WriteLine();

}

static void UnarnUser()

{

var Tr = CreateTriangle();

Console.Write("Результат применения унарной операции: ");

UnarnOp(Tr);

}

static void Unarn()

{

Console.Write("1. Демонстрационный вариант.\n2. Тесты пользователя.\nВаш выбор: ");

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

UnarnOpDemonstr();

break;

case 2:

UnarnUser();

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

static void UnarnOp(Triangle T)

{

if (T.ExistFunc()) Console.WriteLine("Площадь треугольника = " + -T + "\n");

else Console.WriteLine("ТРЕУГОЛЬНИК НЕ СУЩЕСТВУЕТ - ОПЕРАЦИЯ НЕВОЗМОЖНА!\n");

}

//бинарные

static void Binarn(bool ok)

{

Console.Write("1. Демонстрационный вариант.\n2. Тесты пользователя.\nВаш выбор: ");

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

BinarnOpDemonstr(ok);

break;

case 2:

BinarnUser(ok);

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

static void BinarnUser(bool ok)

{

string sign;

if (ok) sign = ">";

else sign = "<";

Console.WriteLine("Создадим первый треугольник:");

var Tr1 = CreateTriangle();

Console.WriteLine("Создадим второй треугольник:");

var Tr2 = CreateTriangle();

Console.Write("Если первый сравним со вторым треугольником знаком '" + sign + "', то получим ответ: ");

BinarnOp(Tr1, Tr2, ok);

}

static void BinarnOpDemonstr(bool ok)

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

string sign;

if (ok) sign = ">";

else sign = "<";

if (ok)

{

Console.WriteLine("2. Бинарная операция '" + sign + "':");

Console.WriteLine("<>Создадим два существующих треугольника(5, 5, 7) и (9, 9, 10):");

var Tr1 = new Triangle(5, 5, 7);

var Tr2 = new Triangle(9, 9, 10);

Console.Write(" Если первый сравним со вторым треугольником знаком '>', то получим ответ: ");

BinarnOp(Tr1, Tr2, true);

Console.WriteLine("<>Создадим два несуществующих треугольника(5, 5, 100) и (9, 9, 109):");

var Tr3 = new Triangle(5, 5, 100);

var Tr4 = new Triangle(9, 9, 109);

Console.Write(" Если первый сравним со вторым треугольником знаком '>', то получим ответ: ");

BinarnOp(Tr3, Tr4, true);

}

else

{

Console.WriteLine("3. Бинарная операция '" + sign + "':");

Console.WriteLine("<>Создадим два существующих треугольника(6, 5, 10) и (8, 9, 13):");

var Tr1 = new Triangle(6, 5, 10);

var Tr2 = new Triangle(8, 9, 13);

Console.Write(" Если первый сравним со вторым треугольником знаком '<', то получим ответ: ");

BinarnOp(Tr1, Tr2, false);

Console.WriteLine("<>Создадим два несуществующих треугольника(34, 5, 100) и (87, 300, 109):");

var Tr3 = new Triangle(34, 5, 100);

var Tr4 = new Triangle(87, 300, 109);

Console.Write(" Если первый сравним со вторым треугольником знаком '<', то получим ответ: ");

BinarnOp(Tr3, Tr4, false);

Console.WriteLine();

}

Console.ResetColor();

}

static void BinarnOp(Triangle Tr1, Triangle Tr2, bool ok)

{

if (Tr1.ExistFunc() && Tr2.ExistFunc())

{

if (ok) Console.WriteLine(Tr1 > Tr2);

else Console.WriteLine(Tr1 < Tr2);

}

else

Console.WriteLine("ТРЕУГОЛЬНИК НЕ СУЩЕСТВУЕТ - ОПЕРАЦИЯ НЕВОЗМОЖНА!");

}

//приведение implicit

static void ImplicitOpDemonstr()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("4. Неявное приведение типа:");

Console.WriteLine("<>Создадим существующий треугольник (7, 10, 11):");

Console.Write(" Сделаем неявное приведение типа и получим периметр: ");

var Tr = new Triangle(7, 10, 11);

ImplicitOp(Tr);

Console.WriteLine("<>Создадим несуществующий треугольник (7, 200, 11):");

Console.Write(" Сделаем неявное приведение типа и получим периметр: ");

var Tr1 = new Triangle(7, 200, 11);

ImplicitOp(Tr1);

Console.ResetColor();

}

static void ImplicitOp(Triangle T)

{

double a = T;

if (T.ExistFunc())

Console.WriteLine(a);

else

Console.WriteLine("ТРЕУГОЛЬНИК НЕ СУЩЕСТВУЕТ - ОПЕРАЦИЯ НЕВОЗМОЖНА.");

}

static void Implicit()

{

Console.Write("1. Демонстрационный вариант.\n2. Тесты пользователя.\nВаш выбор: ");

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

ImplicitOpDemonstr();

break;

case 2:

ImplicitUser();

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

static void ImplicitUser()

{

var Tr = CreateTriangle();

Console.Write("Результат неявного приведения типа(периметр): ");

ImplicitOp(Tr);

}

//приведение Explicit

static void ExplicitOpDemonstr()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("5. Явное приведение типа:");

Console.WriteLine("<>Создадим существующий треугольник (8, 15, 11)");

var Tr = new Triangle(8, 15, 11);

Console.WriteLine(" Сделаем явное приведение типа и поймем, существует ли триугольник: " + Tr.ExistFunc());

Console.WriteLine("<>Создадим несуществующий треугольник (19, 400, 121)");

var Tr1 = new Triangle(19, 400, 121);

Console.WriteLine(" Сделаем явное приведение типа и поймем, существует ли триугольник: " + Tr1.ExistFunc());

Console.ResetColor();

}

static void Explicit()

{

Console.Write("1. Демонстрационный вариант.\n2. Тесты пользователя.\nВаш выбор: ");

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

ExplicitOpDemonstr();

break;

case 2:

ExplicitUser();

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

static void ExplicitUser()

{

var Tr = CreateTriangle();

Console.WriteLine("Результат применения явного приведения типа(существование такого треугольника): " + (bool)Tr + "\n");

}

static void MenuPart3()

{

Console.WriteLine("\n------------------------------МЕНЮ 3-----------------------------------------------------------");

Console.Write("1. Демонстрационная программа.\n2. Создать пустой массив треугольников.\n" +

"3. Создать массив треугольников вручную.\n" +

"4. Создать массив треугольников с помощью ДСЧ.\n" +

"5. Количество корректно созданных треугольников.\n" +

"6. Количество созданных массивов(не null).\n7. Завершить работу.\n" +

"-----------------------------------------------------------------------------------------------" +

"\nВаш выбор: ");

}

static void DemonstrArrOfTriangles()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine(" ДЕМОНСТРАЦИОННАЯ ПРОГРАММА");

Console.WriteLine("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>");

DemonstrEmpty();

Console.WriteLine();

DemonstrNonEmpty();

Console.WriteLine();

DemonstrIndexator();

Console.WriteLine("<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>");

Console.ResetColor();

}

static void DemonstrEmpty()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("1)Создадим объект класса ArrayTriangle, который будет содержать пустой статический массив:");

var arrTr = new ArrayTriangle();

Console.WriteLine("<>Применим функцию, дающую информацию про этот массив: ");

arrTr.ShowArr();

Console.WriteLine("<>Применим функцию, дающую информацию про треугольник с минимальной площадью: ");

arrTr.MinSquare();

Console.ResetColor();

}

static void DemonstrNonEmpty()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("2)Создадим объект класса ArrayTriangle, который будет содержать массив треугольников:");

var arrTr = new ArrayTriangle(true);

Console.WriteLine("<>Применим функцию, дающую информацию про этот массив: ");

arrTr.ShowArr();

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("<>Применим функцию, дающую информацию про треугольник с самой минимальной площадью: ");

arrTr.MinSquare();

Console.ResetColor();

}

static void CreateEmpty()

{

var arrTr = new ArrayTriangle();

Console.WriteLine("<>Применим функцию, дающую информацию про этот массив: ");

arrTr.ShowArr();

Console.WriteLine("<>Применим функцию, дающую информацию про треугольник с минимальной площадью: ");

arrTr.MinSquare();

}

static void CreateArrKeyBoard()

{

Console.Write("Введите количество треугольников, которые вы хотите создать: ");

int i = ParsingInt(Console.ReadLine());

var arrTr = new ArrayTriangle(i, true);

arrTr.MinSquare();

}

static void CreateArrRandom()

{

Console.Write("Введите количество треугольников, которые вы хотите создать: ");

int i = ParsingInt(Console.ReadLine());

var arrTr = new ArrayTriangle(i);

arrTr.ShowArr();

arrTr.MinSquare();

}

static void DemonstrIndexator()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

var TrArr = new ArrayTriangle(true);

var Tr = new Triangle(5, 6, 6);

Console.WriteLine("3)Создадим объект класса ArrayTriangle, который будет содержать массив из 3х треугольников:");

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("<>Присвоим объекту по индексатору TrArr[0] значение другого треугольника(5, 6, 6): " );

TrArr[0] = Tr;

Console.WriteLine();

TrArr[0].Show();

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("<>Присвоим объекту по несуществующему индексатору TrArr[5](всего 3 элемента)" +

"\n значение другого треугольника(5, 6, 6): ");

TrArr[5] = Tr;

TrArr[5].Show();

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("<>Обратимся объекту по индексатору TrArr[0]: ");

TrArr[0].Show();

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine("<>Обратимся к объекту по несуществующему индексатору TrArr[5](всего 3 элемента): ");

TrArr[5].Show();

}

static void Main(string[] args)

{

DemonstrIndexator();

Console.WriteLine(" ВАРИАНТ №10 - КЛАСС 'TRIANGLE'");

Console.WriteLine("\n ЧАСТЬ №1\n");

bool ok = true;

while (ok)

{

MenuPart1();

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

ExistingTriangle();

break;

case 2:

NonExistingTriangle();

break;

case 3:

StaticAndDynamicFunc();

break;

case 4:

CreateTriangle();

break;

case 5:

ZeroAndNegativeSide();

break;

case 6:

Console.WriteLine("\nВсего корректных треугольников было создано: " + Triangle.Count + "\n");

break;

case 7:

ok = false;

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

//Menu2

Console.WriteLine(" ЧАСТЬ №2");

ok = true;

while (ok)

{

MenuPart2();

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

Unarn();

break;

case 2:

Binarn(true);

break;

case 3:

Binarn(false);

break;

case 4:

Implicit();

break;

case 5:

Explicit();

break;

case 6:

Console.WriteLine("\nВсего корректных треугольников было создано: " + Triangle.Count + "\n");

break;

case 7:

ok = false;

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

//Menu3

Console.WriteLine(" ЧАСТЬ №3");

ok = true;

while (ok)

{

MenuPart3();

int i = ParsingInt(Console.ReadLine());

Console.WriteLine();

switch (i)

{

case 1:

DemonstrArrOfTriangles();

break;

case 2:

CreateEmpty();

break;

case 3:

CreateArrKeyBoard();

break;

case 4:

CreateArrRandom();

break;

case 5:

Console.WriteLine("\nВсего корректных треугольников было создано: " + Triangle.Count + "\n");

break;

case 6:

Console.WriteLine("\nВсего массивов(не null) было создано: " + ArrayTriangle.Sum + "\n");

break;

case 7:

ok = false;

break;

default:

Console.WriteLine("Такого пункта нет в меню!");

break;

}

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace лаба9

{

public class ArrayTriangle

{

Triangle[] arr;

public int Size

{

get

{

if (arr == null)

return 0;

return arr.Length;

}

}

private static int sum = 0;

public static int Sum

{

get { return sum; }

}

//индексатор

public Triangle this[int index]

{

get

{

if (index >= 0 && index < Size)

return arr[index];

else

return new Triangle(0, 0, 0);

}

set

{

if (index >= 0 && index < Size)

arr[index] = value;

}

}

//конструкторы

public ArrayTriangle()

{

arr = null;

Console.WriteLine("Создан пустой массив");

}//null

public ArrayTriangle(int count, bool ok)

{

if (count == 0)

{

arr = null;

Console.WriteLine("Создан пустой массив");

}

else

{

arr = new Triangle[count];

for (int i = 0; i < arr.Length; i++)

{

Console.WriteLine($"Треугольник №{i + 1}:");

arr[i] = Program.CreateKeyBoard();

}

sum++;

}

}//вручную

public ArrayTriangle(int count)

{

if (count == 0)

{

arr = null;

Console.WriteLine("Создан пустой массив");

}

else

{

arr = new Triangle[count];

for (int i = 0; i < arr.Length; i++)

{

Console.WriteLine("Треугольник №" + i + ":");

arr[i] = Program.CreateRandom();

}

sum++;

}

}//рандом

public ArrayTriangle(bool ok)

{

arr = new Triangle[3];

arr[0] = new Triangle(5, 4, 7);

arr[1] = new Triangle(19.4, 20.4, 25);

arr[2] = new Triangle(100.4, 95.7, 85.7);

}//пример

//методы

public void ShowArr()

{

if (arr == null) Console.WriteLine("МAССИВ ПУСТ!");

else

{

Console.ResetColor();

for (int i = 0; i < arr.Length; i++)

{

Console.WriteLine($"Треугольник №{i + 1}:");

arr[i].Show();

}

}

}

public void MinSquare()

{

if (arr == null) Console.WriteLine("Найти треугольник с минимальной площадью невозможно, так как массив треугольников пуст!");

else

{

int minIndex = -1;

double minSquare = int.MaxValue;

for (int i = 0; i < arr.Length; i++)

{

if (-arr[i] == 0) Console.WriteLine($"-Площадь треугольника №{ i + 1}: Треугольника не существует; ");

else

{

Console.WriteLine($"-Площадь треугольника №{i + 1}: {-arr[i]};");

if (-arr[i] <= minSquare)

{

minIndex = i;

minSquare = -arr[i];

}

}

}

if (minIndex == -1) Console.WriteLine("Среди треугольников нет существующих - НЕВОЗМОЖНО НАЙТИ МИНИМАЛЬНУЮ ПЛОЩАДЬ!");

else Console.WriteLine($"Треугольник c минимальной площадью {minSquare} стоит на {minIndex + 1} месте.");

}

}

}

}

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using lab9;

namespace Test9

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void ATriangleSideLess0()

{

//arrange

var Tr = new Triangle();

Tr.A = -5;

//act

int x = -5;

//assert

Assert.AreNotEqual(Tr.A, x);

}

[TestMethod]

public void BTriangleSideEqualsTo0()

{

//arrange

var Tr = new Triangle();

Tr.B = 0;

//act

int x = 0;

//assert

Assert.AreNotEqual(Tr.B, x);

}

[TestMethod]

public void CTriangleSideIsOk()

{

//arrange

var Tr = new Triangle();

Tr.C = 4.5;

//act

double x = 4.5;

//assert

Assert.AreEqual(Tr.C, x);

}

[TestMethod]

public void TriangleExistDynam()

{

//arrange

var Tr = new Triangle(4, 5, 6);

bool Ex = Tr.ExistFunc();

//act

bool ok = true;

//assert

Assert.AreEqual(Ex, ok);

}

[TestMethod]

public void TriangleExistStatic()

{

//arrange

var Tr = new Triangle(4, 5, 6);

bool Ex = Triangle.ExistStat(Tr);

//act

bool ok = true;

//assert

Assert.AreEqual(Ex, ok);

}

[TestMethod]

public void TriangleSquare()

{

//arrange

var Tr = new Triangle(3, 4, 5);

double S = -Tr;

//act

double x = 6;

//assert

Assert.AreEqual(S, x);

}

[TestMethod]

public void CompareTrianglesMore()

{

//arrange

var Tr = new Triangle(3, 4, 5);

var Tr1 = new Triangle(6, 8, 10);

bool ok = Tr > Tr1;

//act

bool x = 6 > 24;

//assert

Assert.AreEqual(ok, x);

}

[TestMethod]

public void CompareTriangleLess()

{

//arrange

var Tr = new Triangle(3, 4, 5);

var Tr1 = new Triangle(6, 8, 10);

bool ok = Tr > Tr1;

//act

bool x = 6 > 24;

//assert

Assert.AreEqual(ok, x);

}

[TestMethod]

public void Perimeter()

{

//arrange

var Tr = new Triangle(3, 4, 5);

double p = Tr;

//act

double x = 12;

//assert

Assert.AreEqual(p, x);

}

[TestMethod]

public void OverriddenExistance()

{

//arrange

var Tr = new Triangle(3, 12, 5);

bool p = (bool)Tr;

//act

bool x = false;

//assert

Assert.AreEqual(p, x);

}

[TestMethod]

public void NullArrayOfTriangles()

{

//arrange

var TrArr = new ArrayTriangle();

//act

int x = 0;

//assert

Assert.AreEqual(x, TrArr.Size);

}

[TestMethod]

public void NullArrSize()

{

//arrange

var TrArr = new ArrayTriangle();

int s = TrArr.Size;

//act

int x = 0;

//assert

Assert.AreEqual(x, TrArr.Size);

}

[TestMethod]

public void ArrSize()

{

//arrange

var TrArr = new ArrayTriangle(5);

int s = TrArr.Size;

//act

int x = 5;

//assert

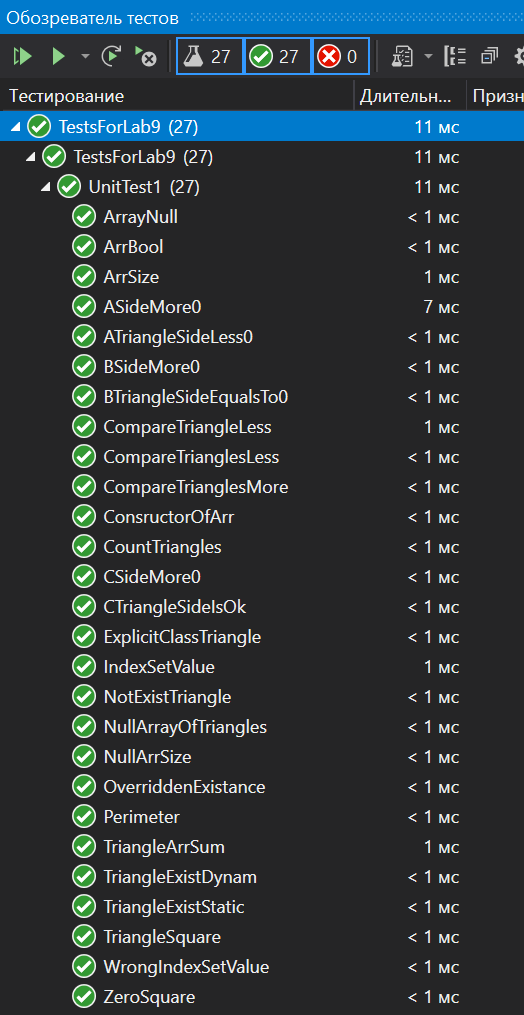
Assert.AreEqual(x, TrArr.Size);

}

}

}

# **Тесты**



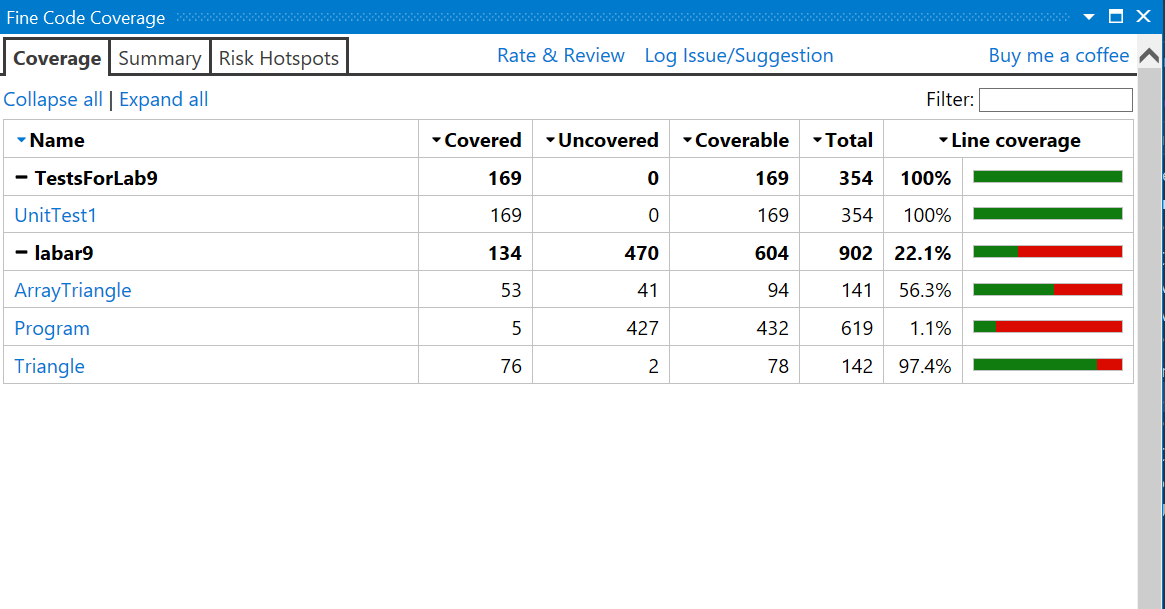


Диаграмма классов:

