Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра Информатики

Дисциплина «Программирование»

**ОТЧЕТ**

к лабораторной работе №7

на тему:

**«Перегрузка Операторов»**

БГУИР 6-05-0612-02 35

|  |
| --- |
| Выполнила студент группы 353504  ГУСЕНЦОВА Екатерина Андреевна |
|  |
| (дата, подпись студента) |
| Проверил ассистент каф. Информатики  РОМАНЮК Максим Валерьевич |
|  |
| (дата, подпись преподавателя) |

Минск 2024

# 1 Индивидуальное задание

**Задание 1. Вариант 5.** Класс полином ax+by+cz. Поля – int (a, b, c). Перегрузить +, -, ++, --, \* на число, / на число. Сравнить на == и !=. (d) если a=b=c=0, полином = false. Преобразовать в число (a) и назад (ax) – в обоих случаях явно.

# 2 Выполнение работы

Язык С# поддерживает перегрузку операторов. За небольшими исключениями большинство операторов С# могут быть перегружены, в результате чего они получат специальное значение по отношению к определённым классам. Для выполнения работы требуется создать класс исходя из индивидуального варианта. Ниже представлен листинг кода файла Polynomial.cs.

using System;

namespace lab7

{

internal class Polynomial

{

private int a;

private int b;

private int c;

public Polynomial(int a, int b, int c)

{

this.a = a;

this.b = b;

this.c = c;

}

public int A

{

get { return a; }

set { a = value; }

}

public int B

{

get { return b; }

set { b = value; }

}

public int C

{

get { return c; }

set { c = value; }

}

public override string ToString()

{

return ("Polynomial = (" + a + "x + " + b + "y + " + c + "z)");

}

public int this[int index]

{

get

{

if (index == 0) return a;

else if (index == 1) return b;

else if (index == 2) return c;

else throw new IndexOutOfRangeException();

}

set

{

if (index == 0) a = value;

else if (index == 1) b = value;

else if (index == 2) c = value;

else throw new IndexOutOfRangeException();

}

}

public static Polynomial operator +(Polynomial p1, Polynomial p2)

{

return new Polynomial(p1.a + p2.a, p1.b + p2.b, p1.c + p2.c);

}

public static Polynomial operator -(Polynomial p1, Polynomial p2)

{

return new Polynomial(p1.a - p2.a, p1.b - p2.b, p1.c - p2.c);

}

public static Polynomial operator \*(Polynomial p, int number)

{

return new Polynomial(p.a \* number, p.b \* number, p.c \* number);

}

public static Polynomial operator /(Polynomial p, int number)

{

if (number == 0)

{

throw new DivideByZeroException();

}

return new Polynomial(p.a / number, p.b / number, p.c / number);

}

public static Polynomial operator ++(Polynomial p)

{

return new Polynomial(p.a + 1, p.b + 1, p.c + 1);

}

public static Polynomial operator --(Polynomial p)

{

return new Polynomial(p.a - 1, p.b - 1, p.c - 1);

}

public static bool operator ==(Polynomial p1, Polynomial p2)

{

return (p1.a == p2.a && p1.b == p2.b && p1.c == p2.c);

}

public static bool operator !=(Polynomial p1, Polynomial p2)

{

return !(p1 == p2);

}

public static bool operator true(Polynomial p)

{

return !(p.a == 0 && p.b == 0 && p.c == 0);

}

public static bool operator false(Polynomial p)

{

return (p.a == 0 && p.b == 0 && p.c == 0);

}

public static explicit operator int(Polynomial p)

{

return p.a+p.b+p.c;

}

public static explicit operator Polynomial(int number)

{

return new Polynomial(number, number, number);

}

}

}

Класс Polynomial имеет конструктор, свойства для доступа к полям a, b, c, метод ToString() для вывода полинома в виде строки, а также индексатор для доступа к компонентам по индексу. Также в классе Polynomial определены перегруженные операторы для выполнения различных операций (+, -, \*, /++, --,==, !=,explicit). В методе Main() происходит демонстрация использования конструктора и свойств, а также работа всех методов и операций. Ниже представлен листинг кода файла Program.cs.

using lab7;

internal class Program

{

static void Main(string[] args)

{

Polynomial poly1 = new Polynomial(5, 9, 3);

Polynomial poly2 = new Polynomial(1, 1, 1);

Console.WriteLine(poly1.ToString());

Console.WriteLine(poly2.ToString() + "\n");

Polynomial sum = poly1 + poly2;

Console.WriteLine("Сумма: " + sum);

Polynomial difference = poly1 - poly2;

Console.WriteLine("Разность: " + difference + "\n");

Polynomial multiplication1 = poly1 \* 2;

Console.WriteLine("Умножение полинома на число: " + multiplication1);

Polynomial delenie = poly1 / (-4);

Console.WriteLine("Деление полинома на число(остается лишь целая часть): " + delenie + "\n");

Polynomial increment = ++poly1;

Console.WriteLine("Инкремент: " + increment);

Polynomial decrement = --poly2;

Console.WriteLine("Декремент: " + decrement + "\n");

bool equal = poly1 == poly2;

Console.WriteLine("Сравнение(==): " + equal);

bool doesNotEqual = poly1 != poly2;

Console.WriteLine("Сравнение(!=): " + doesNotEqual + "\n");

if (poly1)

{

Console.WriteLine("Polynomial не равен (0,0,0).");

}

else

{

Console.WriteLine("Polynomial равен (0,0,0).");

}

if (poly2)

{

Console.WriteLine("Polynomial не равен (0,0,0).");

}

else

{

Console.WriteLine("Polynomial равен (0,0,0).");

}

int convertToInt = (int)poly1;

Console.WriteLine("\nПреобразование в int: " + convertToInt);

Polynomial convertToPolynomialFromInt = (Polynomial)12;

Console.WriteLine("Преобразование в Polynomial: " + convertToPolynomialFromInt + "\n");

}

}

В методе Main() cоздаются два объекта poly1 и poly2 класса Polynomial с заданными компонентами. Выводятся значения векторов на консоль. Выполняются операции сложения, вычитания и умножения полинома на число. Выполняются операции инкремента и декремента полинома. Выполняются операции сравнения полиновом. Проверяется условие истинности полинома (отличность от нулевого полинома). Происходит преобразование полиномов в int и обратно. Результаты операций и преобразований выводятся на консоль.Результат выполнения программы представлен на рисунке 1.

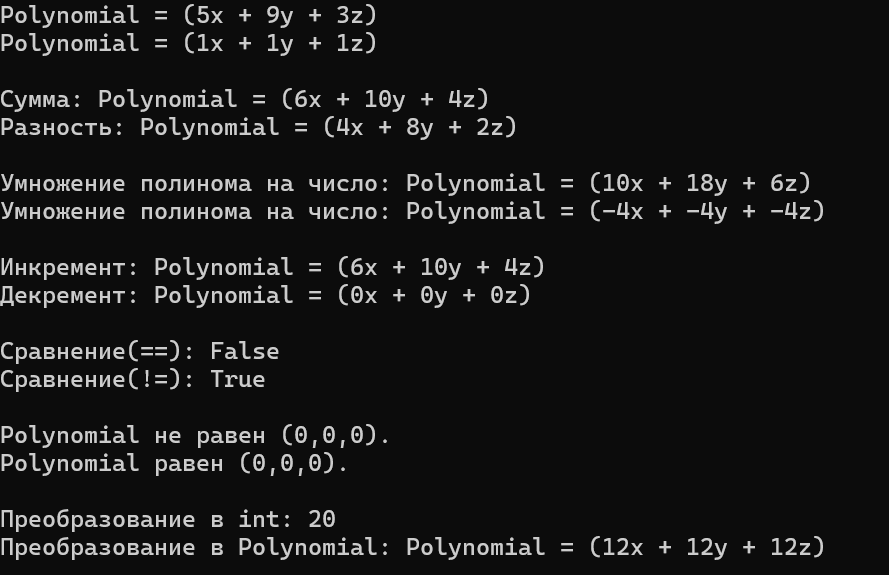


Рисунок 1 – Результат выполнения программы

# Вывод

В ходе лабораторной работы были изучены принципы перегрузки операторов, которые позволяют определить поведение операторов для пользовательских типов данных. были приобретены навыки использования перегрузки различных операторов, таких как арифметические операторы, операторы сравнения и другие. Были изучены результаты их выполнения.