****

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Кафедра «Технология испытаний и эксплуатации»

**Курсовая работа**

Дисциплина «Технологии программирования»

ТЕМА ОБЪЕКТНО ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

КЛАСС «Полином»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

20902 – 01 81 01

Листов 41

Отметка о защите Задание выполнил  
\_\_\_\_\_\_\_ / Молчанова С. И. студент группы Т12О – 209Б – 19

«\_\_» декабря 2020 г.

\_\_\_\_\_\_\_\_ / Атрашков Н. А.

«\_\_» декабря 2020 г.

2020

**Аннотация**

Программа «Полином» является курсовой работой по дисциплине «Технологии программирования» и предназначена для обработки списка массивов параметров легкового автомобиля.

Программа реализуется на платформе интегрированной среды разработки программного обеспечения Microsoft visual studio 2019 community на языке C#.

Пояснительная записка состоит из шести параграфов и двух приложений:

1. Постановка задачи

В данном параграфе представлено задание своего варианта, а также условия и требования для его выполнения.

1. Описание алгоритма

В данном параграфе представлено описание алгоритма программного комплекса «Полином», в котором разобраны необходимые действия для правильной работы программы.

1. Описание классов:

В данном параграфе разобраны все классы программы, приведены описания методов, а также приложена диаграмма классов.

1. Организация данных и диалога

В данном разделе разобрана организация входных и выходных данных программы.

1. Технические и программные средства

В данном параграфе разобраны технические и программные средства, используемые для разработки программы.

1. Результаты работы программы и их оценка

В данном параграфе приводится испытание и тестирование законченной программы.

1. Приложение А. Текст программы

В данном параграфе представлен весь текст программы

1. Приложение Б. Техническое задание на программу

В данном параграфе представлено техническое задание на программу

Оглавление

[**1.** **Постановка задачи** 5](#_Toc58440449)

[**2. Описание алгоритма 6**](#_Toc58440450)

[**3. Описание классов 11**](#_Toc58440451)

[**3.1. Описание методов 11**](#_Toc58440452)

[**3.2. Диаграмма классов 14**](#_Toc58440453)

[**4. Организация данных и диалога 16**](#_Toc58440454)

[**4.1. Организация входных данных 16**](#_Toc58440455)

[**4.2. Организация выходных данных 16**](#_Toc58440456)

[**5. Технические и программные средства 17**](#_Toc58440457)

[**5.1. Технические средства 17**](#_Toc58440458)

[**5.2. Программные средства 17**](#_Toc58440459)

[**6. Результаты работы программы и их оценка 18**](#_Toc58440460)

[**6.1. Объект испытаний 18**](#_Toc58440461)

[**6.2. Цель испытаний 18**](#_Toc58440462)

[**6.3. Требования к программе 18**](#_Toc58440463)

[**6.4. Требования к программной документации 19**](#_Toc58440464)

[**6.5. Методы испытаний 19**](#_Toc58440465)

[**6.6. Оценка результатов тестирования 19**](#_Toc58440466)

[**Заключение 21**](#_Toc58440467)

[**Приложение А. Текст Программы 22**](#_Toc58440468)

[**Приложение Б. Техническое задание 36**](#_Toc58440469)

[**Список используемых источников 41**](#_Toc58440470)

# **Постановка задачи**

Организовать класс «Полином»:

1. Способ передачи параметров в методы - на усмотрение разработчика программы.
2. \*\*Для каждого метода разработать систему кодов возврата, отражающую результат выполнения действий внутри метода, например, успешное выполнение, неправильный номер элемента, неправильный параметр, и т. п.

Требования к программному продукту: при реализации программного продукта необходимо использовать теоретические сведения по дисциплине «Технологии программирования», полученные в процессе изучения дисциплины. При разработке предусматривается наследование специфицированной функциональности, т.е. используется либо абстрактный класс и в этом случае в базовом классе должны присутствовать абстрактные методы, задающие сигнатуру реальных методов, либо на основе интерфейсов, определяющих правила поведения объектов еще не существующих классов.

# **Описание алгоритма**

Для решения поставленной задачи необходимо выполнить следующие действия:

1. определить класс для работы с полиномом;
2. определить размерность массива коэффициентов;
3. организовать считывание коэффициентов из текстового поля;
4. организовать функцию суммы, разности и произведения двух полиномов.

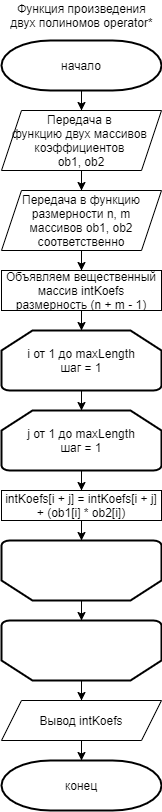


Рисунок 4 - Схема алгоритма произведения двух полиномов

1. **Описание классов**
   1. Описание методов

Ниже представлены описания методов класса Polynomial.

1. Метод ToString() предназначен для преобразование полинома в строку.

* Прототип метода void ToString ();
* Параметры метода отсутствуют;
* Возвращаемое значение str;
* Описание алгоритма метода:
  1. объявить строковую переменную str;
  2. открыть цикл со следующими действиями:
     1. прибавить к str значение коэффициента полинома формата , где:
        + i – порядковый номер элемента полинома;
        + – коэффициент в соответствии с i;
        + – неизвестное число в соответствии с i.
  3. Вернуть str;

На рисунке 1 представлена схема алгоритма преобразования полинома в строку.

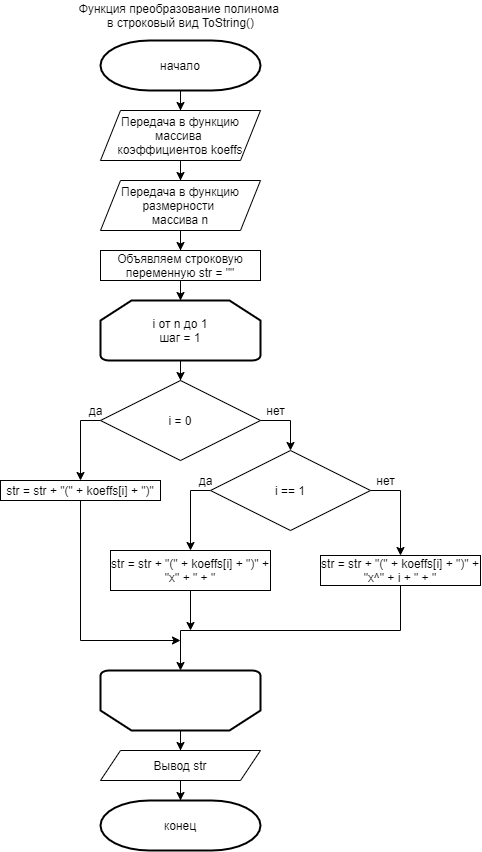


Рисунок 1 – Схема алгоритма преобразования полинома в строку

1. Перегрузка метода operator+() предназначена для перегрузки оператора «+» с целью организации возможности сложения двух полиномов.

* Прототип метода Polynomial operator+();
* Параметры метода:
  1. полином ob1 – первое слагаемое;
  2. полином ob2 – второе слагаемое.
* Возвращаемое значение ob3;
* Описание алгоритма метода:
  1. объявить целочисленные переменные maxLength, minLength, max;
  2. найти среди двух полиномов тот, что обладает максимальной размерностью и присвоить ее значение переменной maxLength, а переменной minLength – значение размерности другого полинома;
  3. присвоить max значение 1, 2 или 0 в случае, если наибольший элемент ob1, ob2 или они равны соответственно;
  4. циклически сложить значения двух полиномов в новый полином ob3;
  5. вывести ob3.

На рисунке 2 представлена схема алгоритма суммы двух полиномов.

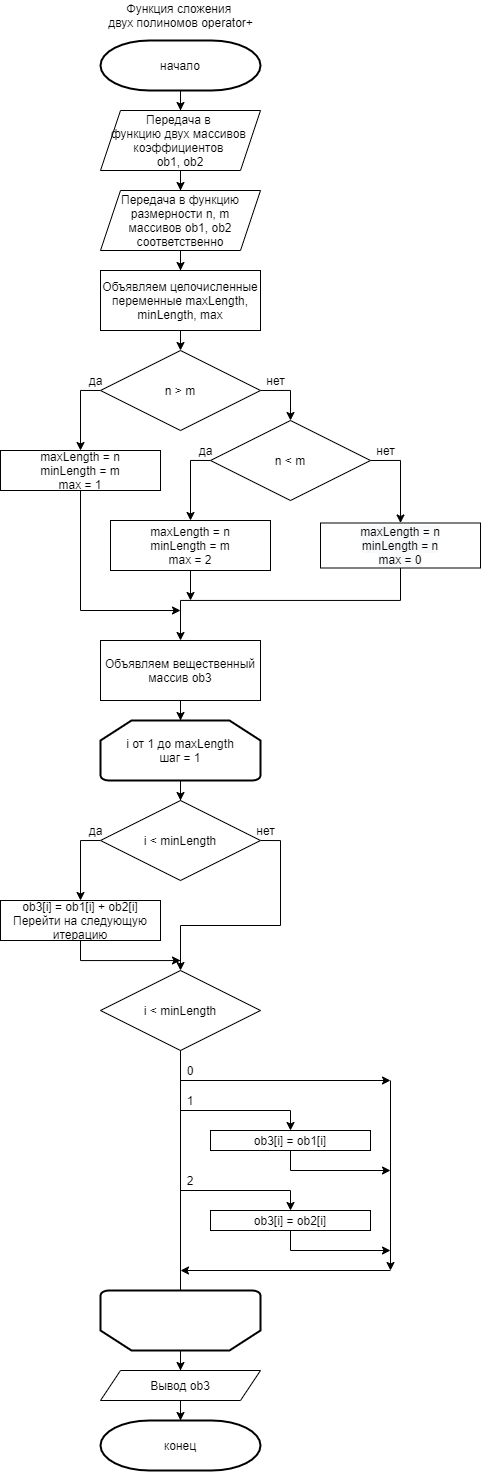


Рисунок 2 - Схема алгоритма суммы двух полиномов

1. Перегрузка метода operator-() предназначена для перегрузки оператора «-» с целью организации возможности вычитания одного полинома из другого.

* Прототип метода Polynomial operator-();
* Параметры метода:
  1. полином ob1 – уменьшаемое;
  2. полином ob2 –вычитаемое.
* Возвращаемое значение ob3;
* Описание алгоритма метода:
  1. объявить целочисленные переменные maxLength, minLength, max;
  2. найти среди двух полиномов тот, что обладает максимальной размерностью и присвоить ее значение переменной maxLength, а переменной minLength – значение размерности другого полинома;
  3. присвоить max значение 1, 2 или 0 в случае, если наибольший элемент ob1, ob2 или они равны соответственно;
  4. циклически найти разность значений двух полиномов в новый полином ob3;
  5. вывести ob3.

На рисунке 3 представлена схема алгоритма разности двух полиномов.

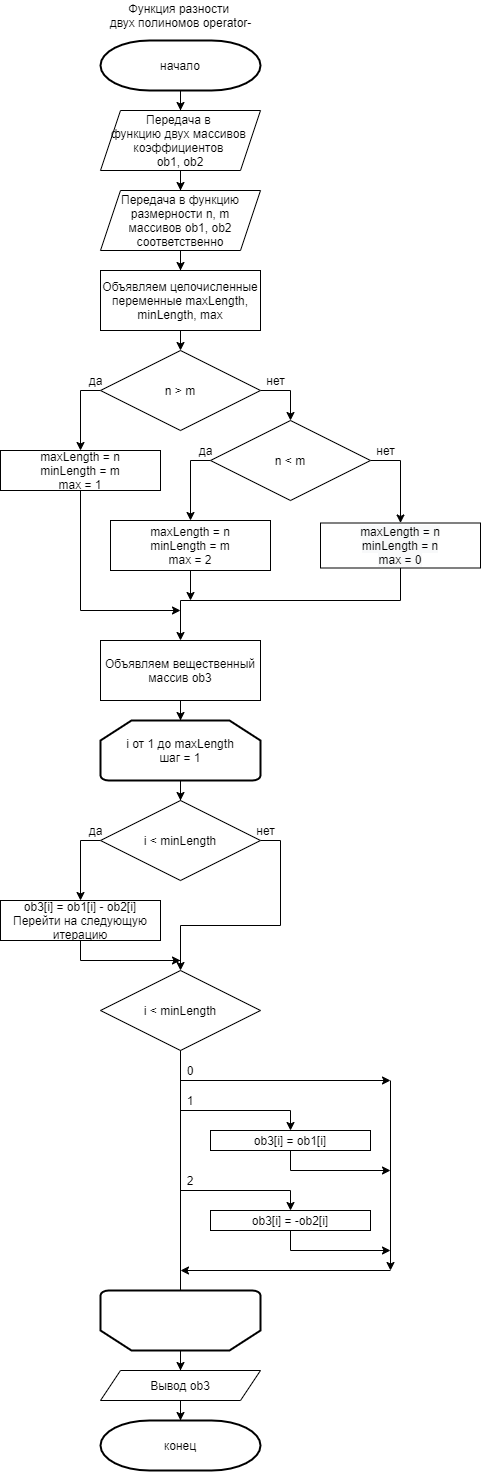


Рисунок 3 - Схема алгоритма разности двух полиномов

1. Перегрузка метода operator\*() предназначена для перегрузки оператора «\*» с целью организации возможности произведения двух полиномов.

* Прототип метода Polynomial operator\*();
* Параметры метода:
  1. полином ob1 – первый множитель;
  2. полином ob2 –второй множитель.
* Возвращаемое значение intKoefs;
* Описание алгоритма метода:
  1. объявить вещественный массив intKoefs размерностью, равной сумме размерностей двух полиномов минус один;
  2. циклически умножить два полинома;
  3. вывести intKoefs.

На рисунке 4 представлена схема алгоритма произведения двух полиномов.

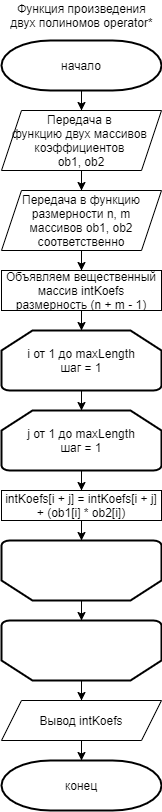


Рисунок 4 - Схема алгоритма произведения двух полиномов

* 1. Диаграмма классов

На рисунке 5 представлена полная диаграмма классов программы.

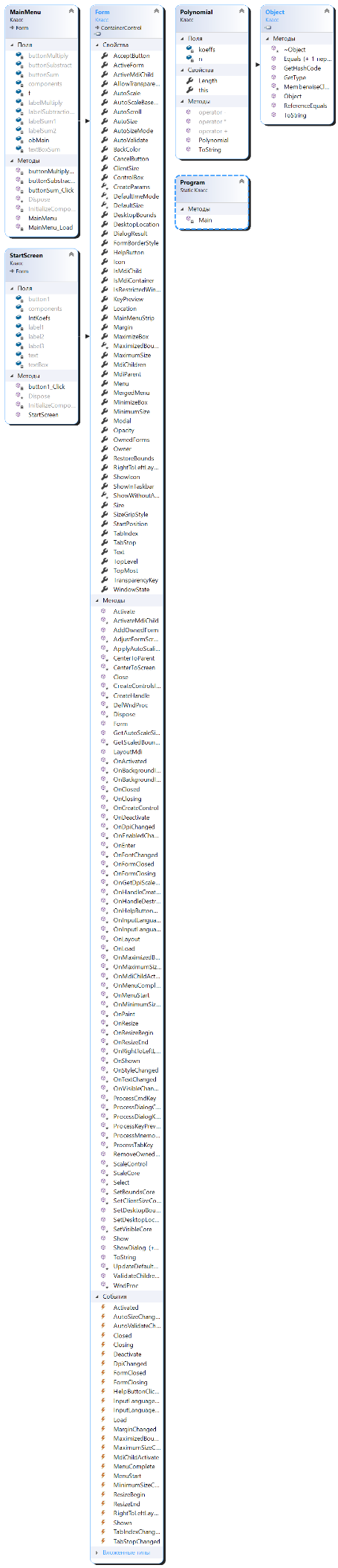


Рисунок 5 – Полная диаграмма классов всей программы

1. **Организация данных и диалога**
   1. Организация входных данных

Входными данными программы являются:

* Полином ob1 – первый вводимый пользователем полином, служащий слагаемым, уменьшаемым или множителем;
* Полином ob2 – второй вводимый пользователем полином, служащий слагаемым, вычитаемым или множителем;
  1. Организация выходных данных

Выходными данными программы являются:

* Полиномы ob1 и ob2, выведенные на экран в строковом формате при помощи функции ToString()
* Информация о работе программы, выведенная на экран.

1. **Технические и программные средства**
   1. Технические средства

При тестировании программы использовался компьютер,

который обладает следующими техническими параметрами:

* процессор с частотой 1,6 ГГц;
* оперативная память объемом 8 Гб;
* 480 ГБ свободного пространства на жестком диске.
  1. Программные средства

При проведении тестов над программой «Полином» использовался компьютер под управлением ОС MS Windows 7.

1. **Результаты работы программы и их оценка**
   1. Объект испытаний

Объектом испытаний является программа «Полином» для работы с полиномами. Программа предназначена для работы с полиномами: нахождение суммы, разности и произведения; а также вывода результатов на экран.

Для проверки работоспособности программы была разработана система тестов, проверяющая основной функционал программной реализации.

* 1. Цель испытаний

Цель испытаний программы «Полином» состоит в том, чтобы оценить работу программы, а именно проверить, соответствует ли характеристики реализованной программы функциональным требованиям, представленным в техническом задании, и перечисленных в разделе «Требования к программе» данного документа. Проверяется корректная работоспособность программной реализации при ей эксплуатации обычным пользователем.

* 1. Требования к программе

Во время проведения испытаний необходимо проверить следующий функционал программы «Полином»:

* сложение двух полиномов;
* разность двух полиномов;
* произведение двух полиномов;
* ввод исходных данных;
* форматированный вывод данных на экран.
  1. Требования к программной документации

Состав программной документации:

* «Полином». Техническое задание.
* «Полином». Пояснительная записка.
  1. Методы испытаний

При тестировании программы использовался метод «чёрного ящика». Суть этого метода заключается в том, что тестировщик не располагает данными о коде программы и тестирует её, только используя её документальное описание и набор входных данных. Такой метод позволяет проводить проверку специалисту, не относящемуся к группе разработчиков, и не требует специальных знаний при проведении тестирования.

* 1. Оценка результатов тестирования

Система тестов программы приведена в таблице 1.

Таблица 1 – Результаты тестирования

|  |  |  |
| --- | --- | --- |
| Назначение | Входные данные | Фактическая реакция |
| Проверка реакции программы на ввод полинома, соответствующему требованиям | 4,9,23.4,59 | Фактическая реакция совпала с ожидаемой |
| Проверка реакции программы на ввод полинома, один или несколько коэффициентов которого не являются числом (целым или вещественным) | 45,a,69,- | Программа запросила пользователя ввести полином заново |
| Проверка реакции программы на ввод полинома, состоящего меньше, чем из двух коэффициентов | 6.56 | Программа запросила пользователя ввести полином заново |
| Проверка реакции программы на ввод «пустого» полинома |  | Программа запросила пользователя ввести полином заново |

# **Заключение**

В процессе разработки программы были полностью освоен интерфейс программирования приложений Windows Forms, при помощи которого удалось создать удобный и интуитивно понятный интерфейс программы.

# **Приложение А. Текст Программы**

Program.cs:

using System;

using System.Windows.Forms;

namespace LPOLYNOMIAL

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Polynomial.cs:

namespace Polynomial

{

class Polynomial

{

double[] koeffs; // Коэфициенты класса

int n; // Количество коэфициентов, определяется само при запуске конструктора

public int Length // Свойство длины полинома

{

get { return koeffs.Length; }

}

public double this[int d] // Индексация массива коэффициентов

{

get { return koeffs[d]; }

set { koeffs[d] = value; }

}

public Polynomial(double[] koeffs)

{

this.koeffs = (double[])koeffs.Clone();

n = this.koeffs.Length;

}

public override string ToString() // Перегрузка "оператора вывода"

{

string str = ""; // Выходная строка

for (int i = Length - 1; i >= 0; i--)

{

if (i == 0)

str += "(" + this[i].ToString() + ")";

else if (i == 1)

str += "(" + this[i].ToString() + ")" + "x" + " + ";

else

str += "(" + this[i].ToString() + ")" + "x^" + i.ToString() + " + ";

}

return str;

}

public static Polynomial operator+(Polynomial ob1, Polynomial ob2) // Сложение полиномов

{

int maxLength; // Максимальная длина

int minLength; // Минимальная длина

int max; // Номер объекта с максимальной длиной

if (ob1.Length > ob2.Length)

{

maxLength = ob1.Length;

minLength = ob2.Length;

max = 1;

}

else if (ob1.Length < ob2.Length)

{

maxLength = ob2.Length;

minLength = ob1.Length;

max = 2;

}

else

{

maxLength = ob1.Length;

minLength = ob1.Length;

max = 0;

}

Polynomial ob3 = new Polynomial(new double[maxLength]);

for (int i = 0; i < maxLength; i++)

{

if (i < minLength)

{

ob3[i] = ob1[i] + ob2[i];

continue;

}

switch (max)

{

case 0:

break;

case 1:

ob3[i] = ob1[i];

break;

case 2:

ob3[i] = ob2[i];

break;

}

}

return ob3;

}

public static Polynomial operator -(Polynomial ob1, Polynomial ob2) // Разность полиномов

{

int maxLength; // Максимальная длина

int minLength; // Минимальная длина

int max; // Номер объекта с максимальной длиной

if (ob1.Length > ob2.Length)

{

maxLength = ob1.Length;

minLength = ob2.Length;

max = 1;

}

else if (ob1.Length < ob2.Length)

{

maxLength = ob2.Length;

minLength = ob1.Length;

max = 2;

}

else

{

maxLength = ob1.Length;

minLength = ob1.Length;

max = 0;

}

Polynomial ob3 = new Polynomial(new double[maxLength]);

for (int i = 0; i < maxLength; i++)

{

if (i < minLength)

{

ob3[i] = ob1[i] - ob2[i];

continue;

}

switch (max)

{

case 0:

break;

case 1:

ob3[i] = ob1[i];

break;

case 2:

ob3[i] = -ob2[i];

break;

}

}

return ob3;

}

public static Polynomial operator \*(Polynomial ob1, Polynomial ob2) // Сложение полиномов

{

double[] intKoefs = new double[ob1.Length + ob2.Length - 1]; // Новый полином

for (int i = 0; i < ob1.Length; ++i)

for (int j = 0; j < ob2.Length; ++j)

intKoefs[i + j] += ob1[i] \* ob2[j];

return new Polynomial(intKoefs);

}

}

}

MainMenu.cs:

using System;

using System.Windows.Forms;

namespace Polynomial

{

public partial class MainMenu : Form

{

StartScreen f;

Polynomial obMain; // Коэфициенты, полученные из предыдущего окна

public MainMenu(StartScreen frm) // Конструктор класса с передачей интерфейса

{

InitializeComponent();

f = frm;

}

private void MainMenu\_Load(object sender, EventArgs e)

{

double[] IntKoefs = (double[])f.IntKoefs.Clone(); // Передача из предыдущей формы значений

obMain = new Polynomial(IntKoefs);

}

private void buttonSum\_Click(object sender, EventArgs e)

{

string[] strKoefs = textBoxSum.Text.Split(','); // Строковые версии коэффициентов

double[] intKoefs = new double[strKoefs.Length]; // Числовые версии коэффициентов

bool ifParsed; // Рабочая; Переманная показывает, что строку правильно спарсили

bool resultParsed = true; // Результат парсинга

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

{

ifParsed = double.TryParse(strKoefs[i], out intKoefs[i]);

if (ifParsed == false)

{

MessageBox.Show("Один или несколько введенных элементов не соответсвуют требованиям", "Ошибка 1", MessageBoxButtons.OK);

resultParsed = false;

break;

}

}

if (intKoefs.Length < 2)

MessageBox.Show("Слишком мало введенных элементов", "Ошибка 2", MessageBoxButtons.OK);

else if (resultParsed == true)

{

Polynomial obSum, obTmp; // Полином суммы и временный полином

obTmp = new Polynomial(intKoefs);

obSum = obMain + obTmp;

MessageBox.Show(obSum.ToString(), "Результат", MessageBoxButtons.OK);

}

}

private void buttonSubstract\_Click(object sender, EventArgs e)

{

string[] strKoefs = textBoxSum.Text.Split(','); // Строковые версии коэффициентов

double[] intKoefs = new double[strKoefs.Length]; // Числовые версии коэффициентов

bool ifParsed; // Рабочая; Переманная показывает, что строку правильно спарсили

bool resultParsed = true; // Результат парсинга

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

{

ifParsed = double.TryParse(strKoefs[i], out intKoefs[i]);

if (ifParsed == false)

{

MessageBox.Show("Один или несколько введенных элементов не соответсвуют требованиям", "Ошибка 1", MessageBoxButtons.OK);

resultParsed = false;

break;

}

}

if (intKoefs.Length < 2)

MessageBox.Show("Слишком мало введенных элементов", "Ошибка 2", MessageBoxButtons.OK);

else if (resultParsed == true)

{

Polynomial obSubstr, obTmp; // Полином разности и временный полином

obTmp = new Polynomial(intKoefs);

obSubstr = obMain - obTmp;

MessageBox.Show(obSubstr.ToString(), "Результат", MessageBoxButtons.OK);

}

}

private void buttonMultiply\_Click(object sender, EventArgs e)

{

string[] strKoefs = textBoxSum.Text.Split(','); // Строковые версии коэффициентов

double[] intKoefs = new double[strKoefs.Length]; // Числовые версии коэффициентов

bool ifParsed; // Рабочая; Переманная показывает, что строку правильно спарсили

bool resultParsed = true; // Результат парсинга

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

{

ifParsed = double.TryParse(strKoefs[i], out intKoefs[i]);

if (ifParsed == false)

{

MessageBox.Show("Один или несколько введенных элементов не соответсвуют требованиям", "Ошибка 1", MessageBoxButtons.OK);

resultParsed = false;

break;

}

}

if (intKoefs.Length < 2)

MessageBox.Show("Слишком мало введенных элементов", "Ошибка 2", MessageBoxButtons.OK);

else if (resultParsed == true)

{

Polynomial obMultipl, obTmp; // Полином разности и временный полином

obTmp = new Polynomial(intKoefs);

obMultipl = obMain \* obTmp;

MessageBox.Show(obMultipl.ToString(), "Результат", MessageBoxButtons.OK);

}

}

}

}

# **Приложение Б. Техническое задание**

****

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Кафедра «Технология Испытаний и Эксплуатации»

**Программный комплекс «Класс Полином»**

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

20902 – 01 90 01

Листов 6

Согласовано Задание принял к исполнению  
\_\_\_\_\_\_\_ / Молчанова С.И. студент группы

«\_\_»\_\_\_\_\_\_\_\_\_\_\_2020 г.

\_\_\_\_\_\_\_\_ /Атрашков Н. А.

«\_\_»\_\_\_\_\_\_\_\_\_\_\_2020 г.

2020

**Введение**

Программный комплекс «Полином» (далее по тексту POLYNOMIAL) может быть использован для решения задач в области прикладной математики.

1. **Основание для разработки**

Основанием для разработки является учебный план по дисциплине «Технологии программирования».

1. **Назначение разработки**

Программа POLYNOMIAL является курсовой работой по дисциплине «Технологии программирования» и предназначается для выполнения алгебраических операций с полиномами.

1. **Требования к программе**

**3.1. Требования к функциональным характеристикам**

Программа POLYNOMIAL должна выполнять следующие функции:

* организацию меню;
* вывод сведений об авторе;
* сложение двух полиномов;
* разность двух полиномов;
* произведение двух полиномов;
* ввод исходных данных;
* форматированный вывод данных на экран;
* формирование отчета о работе программы;
* создание интуитивно понятного интерфейса.
  1. **Организация входных и выходных данных**

Организация входных данных

Входными данными для программы являются:

* целое положительное число – ключ меню;
* целые положительные числа – коэффициенты полинома;
* действительные числа, являющиеся коэффициентами полинома.

Организация выходных данных

Выходными данными для программы являются:

* текстовый файл, содержащий сохраненные полиномы;
* перечень диагностических сообщений;
* отчет о работе программы.
  1. **Требования к надёжности**  
      Для обеспечения надежного функционирования программы необходимо предусмотреть проверку введенных данных, ключей меню и выдачу диагностических сообщений при обнаружении ошибок.
  2. **Условия эксплуатации**

Для эксплуатации программы не требуется специально подготовленный пользователь.

* 1. **Требования к составу и параметрам технических средств**

Для функционирования программы необходимо наличие следующих технических средств:

* процессор Intel® Celeron™;
* не менее 256 Мб оперативной памяти;
* монитор SVGA с минимальным разрешением 800х600 пикселей;
* манипулятор типа «мышь» и клавиатура.

**3.6. Требования к информационной и программной совместимости**

Для разработки программы POLYNOMIAL?? необходим следующий инструментарий:

* MS VisualStudio Community 2019 – платформа выбрана для использования языка С#;
* операционная система MS Windows 7 или выше.

1. **Требования к программной документации**

Состав программной документации

* «Программа POLYNOMIAL». Техническое задание.
* «Программа POLYNOMIAL». Пояснительная записка.

1. **Стадии и этапы разработки**

**5.1. Эскизный проект**

Обзор существующих программных решений.

Предварительная разработка структуры входных и выходных данных.

Разработка общего описания алгоритма.

Срок окончания: 9-10 недели

**5.2. Технический проект**

Уточнение структуры входных и выходных данных.

Разработка средств обработки ошибок и выдачи диагностических сообщений.

Окончательное определение конфигурации технических и программных средств.

Разработка пояснительной записки.

Срок окончания: 11-12 недели

* 1. **Рабочий проект**

Написание программы.

Тестирование и отладка программы.

Разработка программной документации.

Разработка программы и методики испытаний.

Корректировка программы и программной документации по результатам испытаний.

Срок окончания:13-16 недели

* 1. **Внедрение**

Защита курсовой работы.

Срок окончания:17-18 недели

1. **Порядок контроля и приёмки**

Контроль и приёмка разработки осуществляется в соответствии с программой и методикой испытаний, изложенными в пояснительной записке.

1. **Перечень сокращений**

SVGA – Super Video Graphics Array – улучшенный видеографический адаптер, обеспечивающий разрешающую способность 1280х1024 точек и более чем 256 цветов.

Мб – Мегабайт – единица измерения объема информации.

1. **Перечень используемой литературы**

* Курс лекций по дисциплине «Технология программирования» Молчанова С. И.
* Шилдт Герберт - Справочник программиста по С/С++, 3-е изд.: Пер. с англ. – М.: Издательский дом “Вильямс”, 2003. – 432 с.
* Герберт Шилдт - C# 4.0 Полное руководство: Пер. с англ. – М. : Издательский дом “Вильямс”, 2011. – 1056 с.
* Руководство по C# от Microsoft [Электронный ресурс] - https://docs.microsoft.com/
* Самоучитель по C# [Электронный ресурс] - https://nullpro.info/
* Форум по программированию [Электронный ресурс] - https://stackoverflow.com/

# **Список используемых источников**

* Курс лекций по дисциплине «Технология программирования» Молчанова С. И.
* Шилдт Герберт - Справочник программиста по С/С++, 3-е изд.: Пер. с англ. – М.: Издательский дом “Вильямс”, 2003. – 432 с.
* Герберт Шилдт - C# 4.0 Полное руководство: Пер. с англ. – М. : Издательский дом “Вильямс”, 2011. – 1056 с.
* Руководство по C# от Microsoft [Электронный ресурс] - https://docs.microsoft.com/
* Самоучитель по C# [Электронный ресурс] - https://nullpro.info/
* Форум по программированию [Электронный ресурс] - https://stackoverflow.com/