**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

**отчет**

**по лабораторной работе №4**

**по дисциплине «Оптимальное проектирование»**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3301 |  | Малинина Е.А. |
| Преподаватель |  | Каримов А.И. |

Санкт-Петербург

2017

**Лабораторная работа №4**   
Проектирование пользовательского интерфейса

Цель работы

Изучение современной технологии разработки диалоговых комплексов оптимизации, ориентированной на применение языков программирования C#; модификация программы, разработанной в предыдущих лабораторных работах.

Задание

Модифицированная программа должна обеспечивать:

* оптимизацию целевых функций с *любым* числом переменных;
* Программа должна быть дружественной пользователю:
* содержать необходимые заставки и сообщения, характеризующие программу, используемый метод, характер выводимой информации, допущенную ошибку при вводе данных и т. д.;
* содержать блокировку ошибочных действий при вводе данных и обеспечивать простоту исправления ошибки;
* обеспечивать в демонстрационном режиме возможность выбора одной из нескольких (трех-пяти) тестовых функций и запуск на поиск решения.

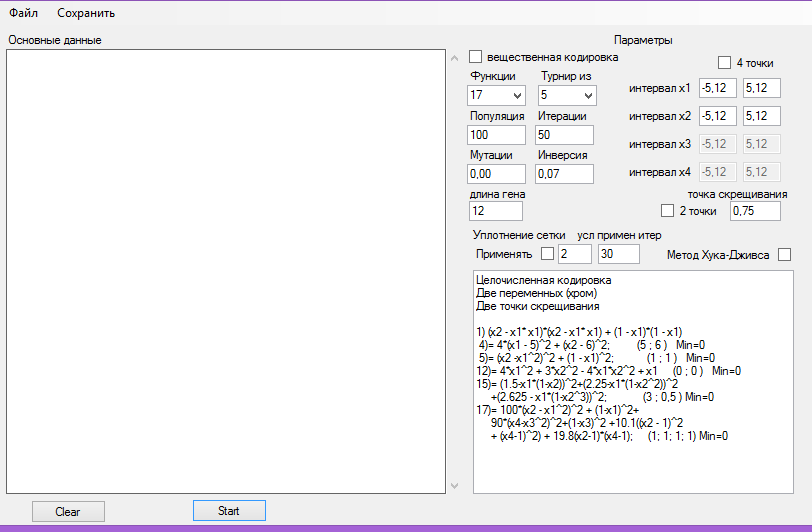
При составлении программы предусмотреть:

* структурирование программы в соответствии с объектно-ориентированным подходом к разработке программного обеспечения;
* использование класса битовой цепочки для представления в программе операций для работы с хромосомами;
* возможность сравнения по качеству найденного решения двух способов кодирования хромосом – целочисленного и вещественного;
* выбор по требованию пользователя способа кодирования хромосом, возможность смены типа скрещивания. Также предусмотреть возможность опционального отключения дополнительных операторов (популяционного всплеска и уплотнения сетки);
* сохранение журналов с подробным описанием хода решения и значениями ведущих переменных в текстовый файл.
* Текст программы с детальными комментариями ведущих операторов программы.
* Описание интерфейса пользователя программы.

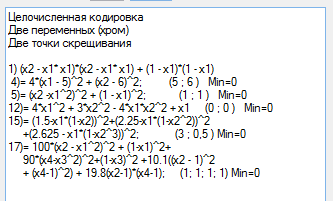
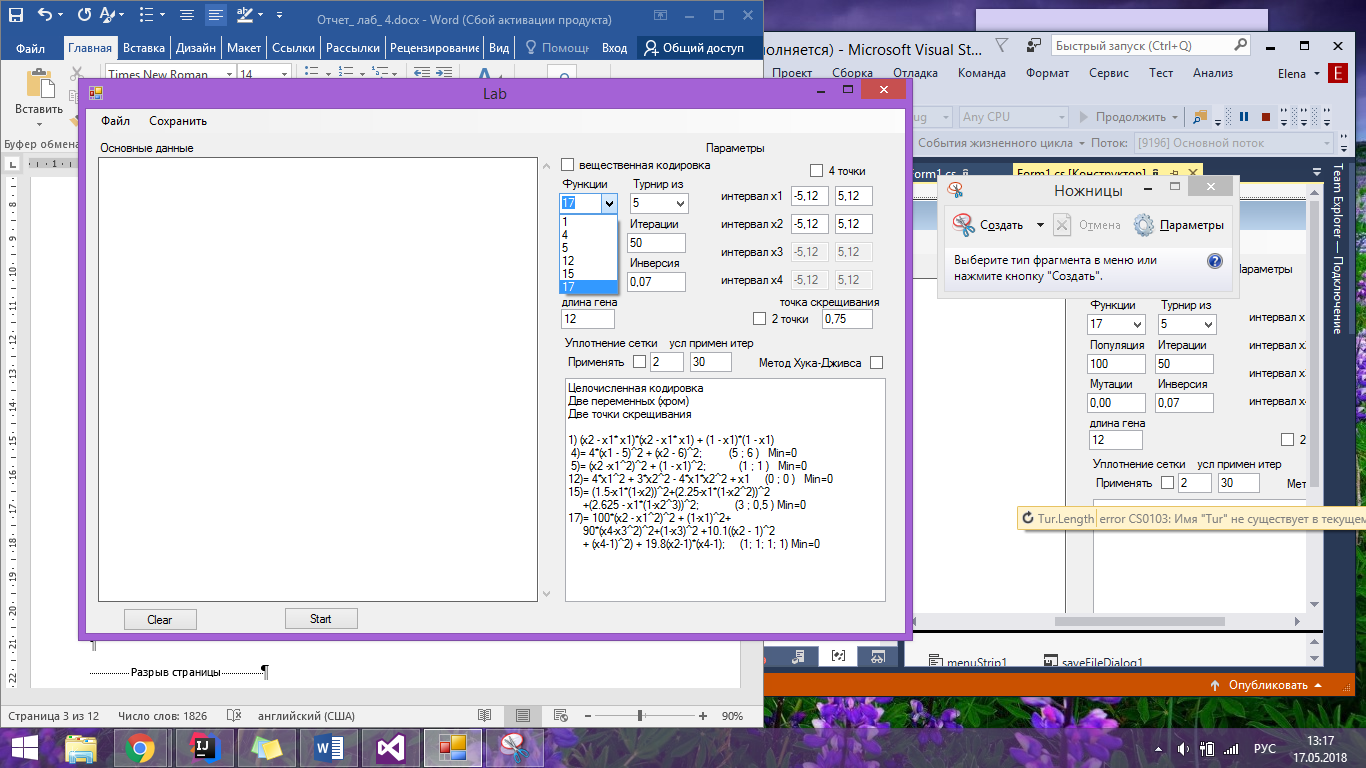
Спецификация

|  |  |  |
| --- | --- | --- |
| **Класс** | **Назначение** | |
| class Chromosom | Работает с целочисленной хромосомой | |
| class Individ | Работает с особью содержащий целочисленную хромосому | |
| class Population | Работает с несколькими особями | |
| class VeshChromosom | Работает с вещественной хромосомой | |
| class VeshIndivid | Работает с особью содержащий вещественною хромосому | |
| class VeshPopulation | Работает с несколькими особями | |
| class Parametrs | Содержит все переменные-параметры алгоритма | |
| **Функция** | | **Назначение** |
| private void Int\_Nomb(object sender, KeyPressEventArgs e) | | Ограничения по вводу. Вводятся только цифры и запятая |
| private void сохранить(object sender, EventArgs e) | | Сохранение результатов тестирования |

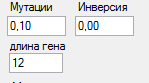
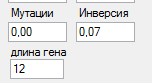
Интерфейс программы



Возможность выбора функций. Они заранее представлены в нижнем вспомогательном окне.



Так же есть возможность выбора количества особей, участвующих в турнирном отборе, от 2 до 5.



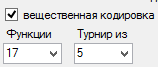
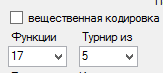
Значение мутации и инверсии могут быть изменены пользователем вручную. Длина гена так же может быть изменена пользователем.

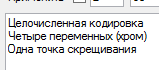
 

При работе с функциями от двух переменных активны два интервала х1 и х2. Для работы с функциями от 4 переменных необходимо нажать галочку в верхнем поле и проверить верность заданных промежутков. Все окна для ввода позволяют ввести только цифры и запятую.



Точка скрещивания задается в процентах от длины гена. Так же может быть изменена пользователем вручную. При нажатии галочки «2 точки» будет проходить двухточечное скрещивание. При этом программа сделает две зеркальных точки, например, одна точка будет 0,25 другая 0,75. При любом значении программа автоматический сделает первую точку меньше второй.





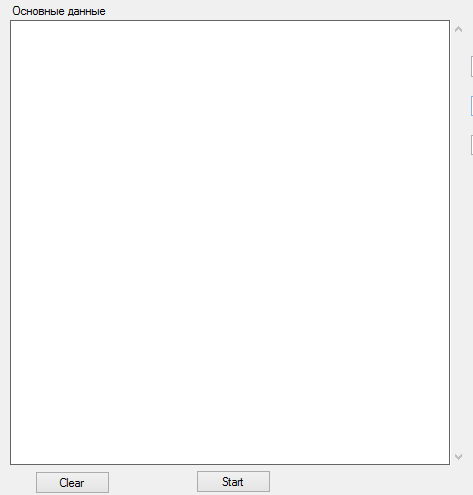
Для смены целочисленной кодировки на вещественную достаточно только поставить галочку у кнопки «вещественная кодировка». Так же в дополнительном окне отображается текущая кодировка.



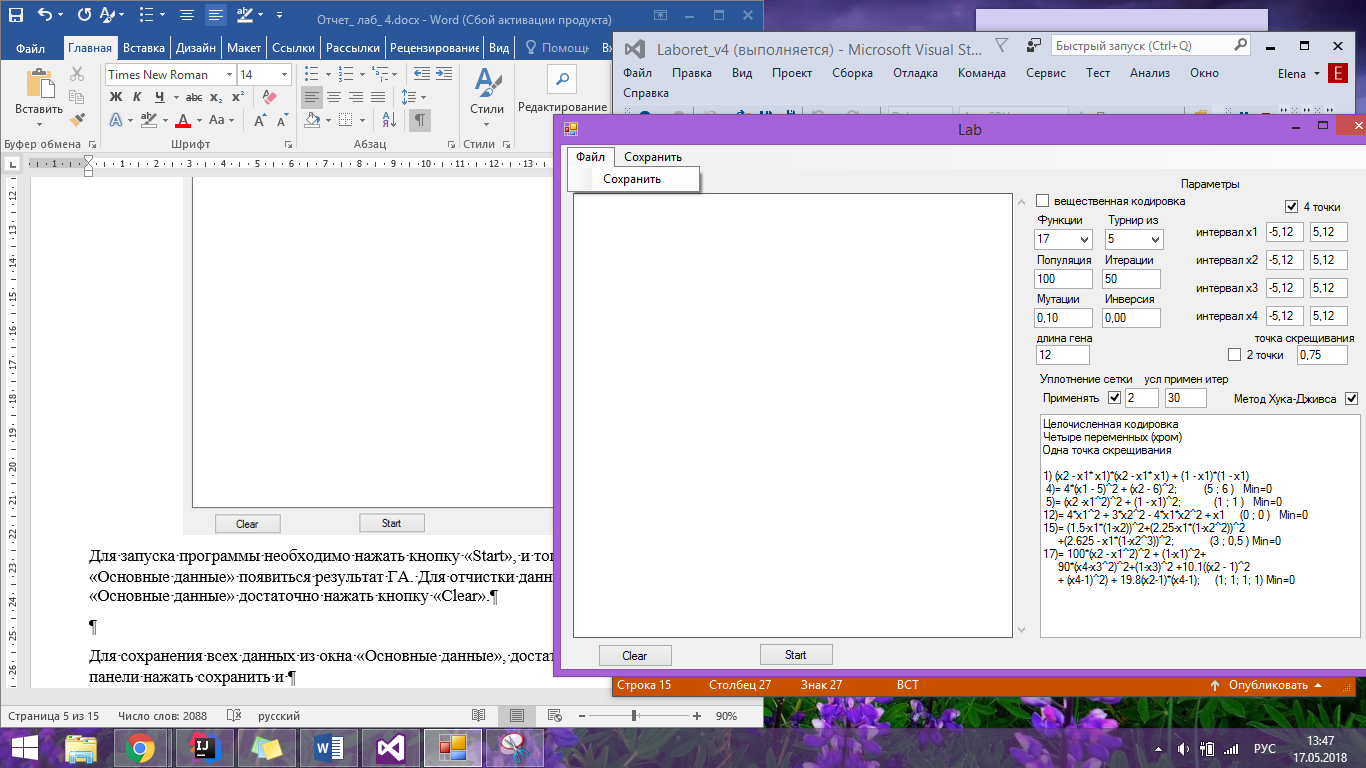
Уплотнение сетки будет происходить, когда будет галочка при кнопке «Применять». Первое окно задает во сколько раз будет уплотнена сетка. На приведенных изображениях сетка будет уплотнена в 2 раза. Следующее окно показывает с какой итерации начнется уплотнение, ни изображениях с 30 популяции. Уплотнение сетки и условие применения уплотнения сетки можно поменять вручную.



Метод Хука-Дживса применяется после генетического алгоритма. И его применение после ГА регулируется галочкой около названия метода.

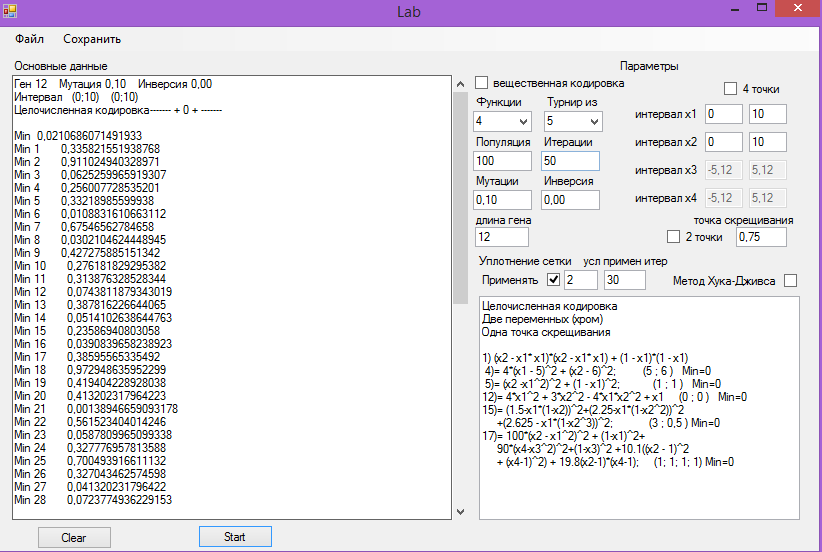


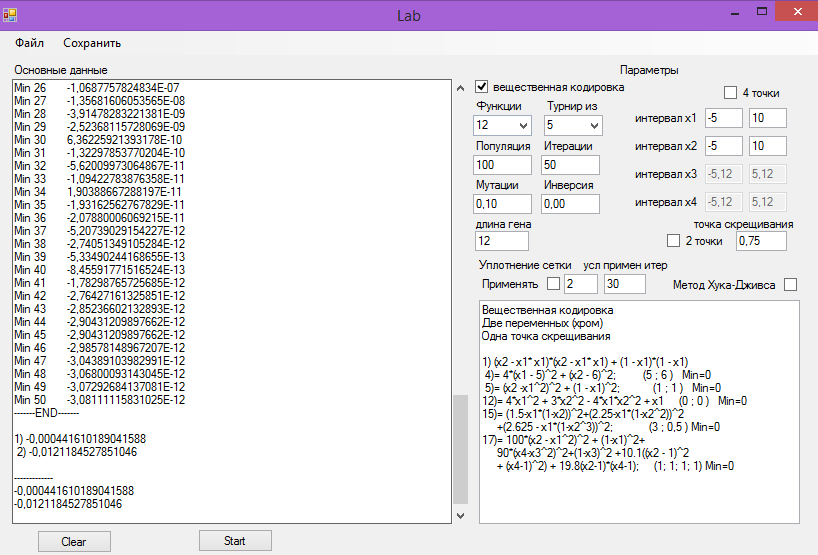
Для запуска программы необходимо нажать кнопку «Start», и тогда в окне «Основные данные» появиться результат ГА. Для отчистки данных в окне «Основные данные» достаточно нажать кнопку «Clear».

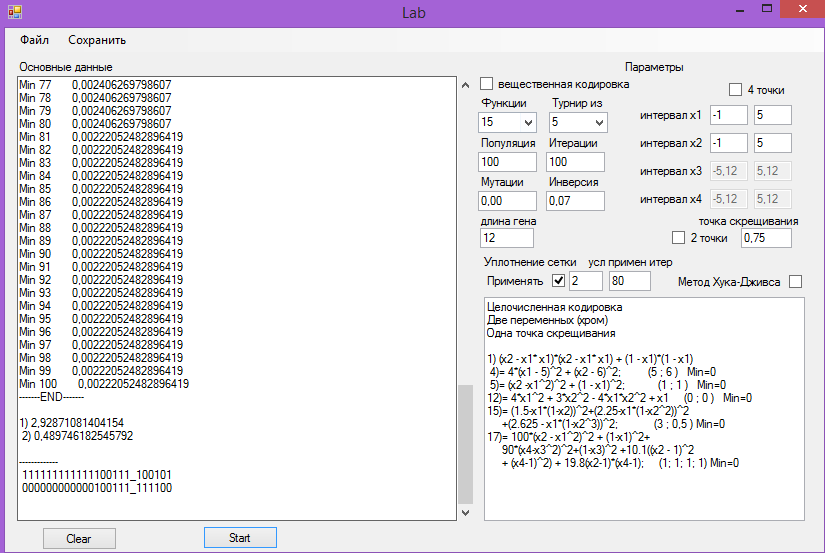


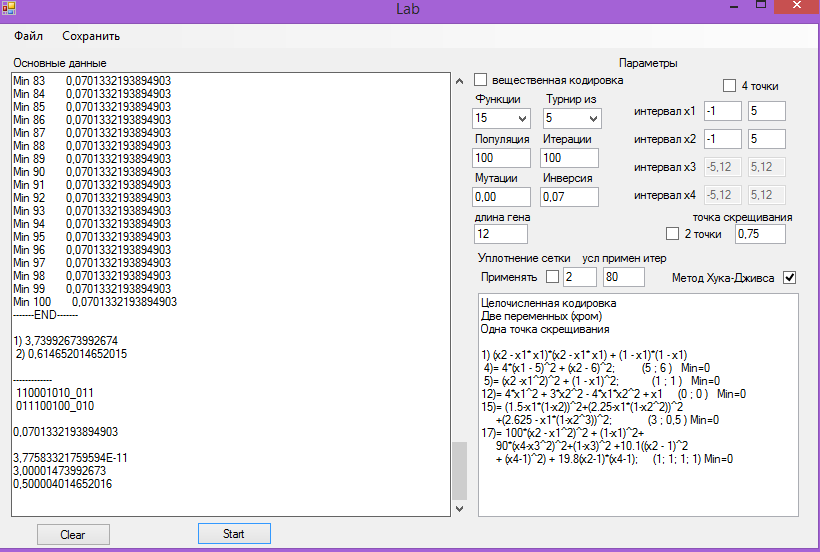
Для сохранения всех данных из окна «Основные данные», достаточно в верхней панели нажать сохранить и в выплывающем окне сохранить данные в .txt формате.

**Тестирование программы**









Вывод

В ходе проделанной работы ознакомились с современными технологиями разработки диалоговых комплексов оптимизации, ориентированных на применение языков программирования C#. Для рационализации выполнения лабораторных работ эту лабораторную надо делать первой, и постепенно с каждой следующей дополнять ее интерфейс. Так как если пытаться к созданной программе присоединять интерфейс намного сложнее, нежели сделать его заранее и как в ГА позволять ему эволюционировать вместе с сложностью лабораторных работ.

Код программы

public partial class Form1 : Form

{

Parametrs Aa = new Parametrs();

double x1, x2;public Form1()

{

InitializeComponent();

help\_box();

}

private void Int\_Nomb(object sender, KeyPressEventArgs e)

{ char number = e.KeyChar;

if (!Char.IsDigit(number) && number != 8 && number != 44 && number != 45) // цифры, клавиша BackSpace и запятая

{

e.Handled = true;

}

}

private void Mutation\_KeyPress(object sender, KeyPressEventArgs e)

{ Int\_Nomb(sender, e); }

private void Inversion\_KeyPress(object sender, KeyPressEventArgs e)

{ Int\_Nomb(sender, e); }

private void VerPoint\_KeyPress(object sender, KeyPressEventArgs e)

{ Int\_Nomb(sender, e); }

private void сохранитьToolStripMenuItem1\_Click(object sender, EventArgs e)

{

сохранить( sender, e);

}

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

{

сохранить(sender, e);

}

private void сохранить(object sender, EventArgs e)

{

SaveFileDialog saveFileDialog1 = new SaveFileDialog();

saveFileDialog1.Filter = "txt files (\*.txt)|\*.txt";

saveFileDialog1.RestoreDirectory = true;

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{string filename = saveFileDialog1.FileName;

System.IO.File.WriteAllText(filename, MtextBox.Text);

}

}

///////////////////////////////////////////////////////////////////////////

private void button1\_Click(object sender, EventArgs e)

{

Set\_param();

// Test\_Chrom();

// Test\_Individ();

//

//Test\_VeshCr();

// Test\_VeshInd();

if (Aa.Kod)

Test\_VeshPopultion();

else

if (!Aa.Set) Test\_Popultion();

else Test\_Popultion2();

if (checkBox5.Checked)

XD();// метод хука ждивса

/// checkBox5 метод ХД

}

//вещественная кодировка

private void Test\_VeshPopultion()

{

VeshPopulation test = new VeshPopulation(Aa);

// создание популяции

//int Namber0, int nIters0, double pMut0, double pInv0

// ( (int L, float Mut, float Inv, int point, int point2, int Namber0, int nIters0)

VeshIndivid test\_Ind = test.Turnir(test.Popul\_all);

// MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test.Print( point, point2) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Fank\_out + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen(Aa) + Environment.NewLine;

MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine

// + test.Print(Aa) //;

+ Environment.NewLine + "Min " + test\_Ind.Fank\_out + Environment.NewLine;

for (int i = 0; i < Aa.Itertion; i++)

{

test.Select(Aa);

test\_Ind = test.Turnir(test.Popul\_all);

MtextBox.Text = MtextBox.Text + //"-------" + (i+1) + "-------" + Environment.NewLine +

// test.Print(point, point2)+ Environment.NewLine + "-----" + Environment.NewLine +

// test\_Ind.Fank\_out + Environment.NewLine;

// test.Print(Aa) + Environment.NewLine +

"Min " + (i + 1) + " " + test\_Ind.Fank\_out + Environment.NewLine;

test.Mut\_all(Aa);

//MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

}

//test\_Ind = test.Turnir(test.Popul\_all);

MtextBox.Text = MtextBox.Text + "-------END-------" + Environment.NewLine +

// test.Print(point, point2) + Environment.NewLine + "-----" +

Environment.NewLine + test\_Ind.Prin\_Fen(Aa) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

MtextBox.Text = MtextBox.Text + "-------------" + Environment.NewLine +

test\_Ind.xrom[0].Prin() + Environment.NewLine + test\_Ind.xrom[1].Prin() + Environment.NewLine;

/////для 4

x1 = test\_Ind.xrom[0].chrom;

x2 = test\_Ind.xrom[1].chrom;

}

// целочисленная кодировка

private void Test\_Popultion()

{

Population test = new Population(Aa);

// создание популяции

//int Namber0, int nIters0, double pMut0, double pInv0

// ( (int L, float Mut, float Inv, int point, int point2, int Namber0, int nIters0)

Individ test\_Ind = test.Turnir2(test.Popul\_all);

// MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test.Print( point, point2) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Fank\_out + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen(Aa) + Environment.NewLine;

Pr\_param();

MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine

// + test.Print(Aa) //;

+ Environment.NewLine+ "Min " + test\_Ind.Fank\_out+Environment.NewLine;

for (int i = 0; i < Aa.Itertion; i++)

{

test.Mut\_all(Aa);

test.Select(Aa);

test\_Ind = test.Turnir2(test.Popul\_all);

MtextBox.Text = MtextBox.Text + //"-------" + (i+1) + "-------" + Environment.NewLine +

// test.Print(point, point2)+ Environment.NewLine + "-----" + Environment.NewLine +

// test\_Ind.Fank\_out + Environment.NewLine;

// test.Print(Aa) + Environment.NewLine +

"Min " + (i + 1) +" " + test\_Ind.Fank\_out + Environment.NewLine;

test.Mut\_all(Aa);

//MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

}

//test\_Ind = test.Turnir(test.Popul\_all);

MtextBox.Text = MtextBox.Text + "-------END-------" + Environment.NewLine +

// test.Print(point, point2) + Environment.NewLine + "-----" +

Environment.NewLine + test\_Ind.Prin\_Fen(Aa) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

MtextBox.Text = MtextBox.Text + "-------------" + Environment.NewLine +

test\_Ind.xrom[0].Prin(Aa) + Environment.NewLine + test\_Ind.xrom[1].Prin(Aa) + Environment.NewLine;

/////для 4

x1 = test\_Ind.Fenotip\_1(Aa);

x2 = test\_Ind.Fenotip\_2(Aa);

}

// целочисленная кодировка с уплотнением сетки

private void Test\_Popultion2()

{

Population test = new Population(Aa);

// создание популяции

int i;

//int Namber0, int nIters0, double pMut0, double pInv0

// ( (int L, float Mut, float Inv, int point, int point2, int Namber0, int nIters0)

Individ test\_Ind = test.Turnir2(test.Popul\_all);

// MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test.Print( point, point2) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Fank\_out + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen(Aa) + Environment.NewLine;

Pr\_param();

MtextBox.Text = MtextBox.Text + "------- + 0 + -------" + Environment.NewLine

// + test.Print(Aa) //;

+ Environment.NewLine + "Min " + test\_Ind.Fank\_out + Environment.NewLine;

for (i = 0; i < Aa.DopIter; i++)

{

test.Mut\_all(Aa);

test.Select(Aa);

test\_Ind = test.Turnir2(test.Popul\_all);

MtextBox.Text = MtextBox.Text + //"-------" + (i+1) + "-------" + Environment.NewLine +

// test.Print(point, point2)+ Environment.NewLine + "-----" + Environment.NewLine +

// test\_Ind.Fank\_out + Environment.NewLine;

// test.Print(Aa) + Environment.NewLine +

"Min " + (i + 1) + " " + test\_Ind.Fank\_out + Environment.NewLine;

test.Mut\_all(Aa);

//MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

}

Aa.Length = Aa.Length \* Aa.Setka;

Population test2 = new Population(Aa);

test2.Copi(Aa, test);

Individ test\_Ind2 = test2.Turnir2(test2.Popul\_all);

Aa.point1 = Aa.point1\* Aa.Setka;

Aa.point2 = Aa.point2\* Aa.Setka;

for (i = Aa.DopIter; i < Aa.Itertion; i++)

{

test2.Mut\_all(Aa);

test2.Select(Aa);

test\_Ind2 = test2.Turnir2(test2.Popul\_all);

MtextBox.Text = MtextBox.Text + //"-------" + (i+1) + "-------" + Environment.NewLine +

// test.Print(point, point2)+ Environment.NewLine + "-----" + Environment.NewLine +

// test\_Ind.Fank\_out + Environment.NewLine;

// test.Print(Aa) + Environment.NewLine +

"Min " + (i + 1) + " " + test\_Ind2.Fank\_out + Environment.NewLine;

test2.Mut\_all(Aa);

//MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

}

//test\_Ind = test.Turnir(test.Popul\_all);

MtextBox.Text = MtextBox.Text + "-------END-------" + Environment.NewLine +

// test.Print(point, point2) + Environment.NewLine + "-----" +

Environment.NewLine + test\_Ind2.Prin\_Fen(Aa) + Environment.NewLine;

// MtextBox.Text = MtextBox.Text + test\_Ind.Prin\_Fen() + Environment.NewLine;

MtextBox.Text = MtextBox.Text + "-------------" + Environment.NewLine +

test\_Ind2.xrom[0].Prin(Aa) + Environment.NewLine + test\_Ind2.xrom[1].Prin(Aa) + Environment.NewLine;

/////для 4

x1 = test\_Ind.Fenotip\_1(Aa);

x2 = test\_Ind.Fenotip\_2(Aa);

}

// установление всех параметров

private void Pr\_param()

{

MtextBox.Text = MtextBox.Text + "Ген " + LengthGene.Text +

" Мутация " + Mutation.Text + " Инверсия " + Inversion.Text + Environment.NewLine;

if (Aa.Chom == 2)

MtextBox.Text = MtextBox.Text + "Интервал "+

" (" + min\_x1.Text + ";" + max\_x1.Text + ") " +

" (" + min\_x2.Text + ";" + max\_x2.Text + ") " + Environment.NewLine;

else

MtextBox.Text = MtextBox.Text + "Интервал " +

" (" + min\_x1.Text + ";" + max\_x1.Text + ") " +

" (" + min\_x2.Text + ";" + max\_x2.Text + ") " + Environment.NewLine +

" (" + min\_x3.Text + ";" + max\_x3.Text + ") " +

" (" + min\_x4.Text + ";" + max\_x4.Text + ") " + Environment.NewLine;

if (Aa.Kod)

MtextBox.Text = MtextBox.Text+"Вещественная кодировка";

else

MtextBox.Text = MtextBox.Text+"Целочисленная кодировка";

}

private void Set\_param()

{

Aa.Length = Convert.ToInt32(LengthGene.Text); // длина гена

Aa.Muttion = float.Parse(Mutation.Text); // мутация

Aa.Invtion = float.Parse(Inversion.Text); // инверсия

//проверка

if ( Convert.ToInt32(Population.Text) % 2 > 0)

Aa.Popultion = Convert.ToInt32(Population.Text)+1; // количество особей популяции

else Aa.Popultion = Convert.ToInt32(Population.Text);

//проверка

if (Aa.Popultion < 10)

Aa.Tutu= 2; // количество особей популяции

else

Aa.Tutu = Convert.ToInt32(comboBox1.Text);

Aa.Itertion = Convert.ToInt32(Iteration.Text); // кол итераций

Aa.Fanc = Convert.ToInt32(fanction\_Ch.Text); // функция

//MtextBox.Text = MtextBox.Text + fanction\_Ch.Text;

// поисковый интервал

// количество Х

if (Aa.Chom == 2)

Aa.interM = new float[,] { {float.Parse(min\_x1.Text) ,float.Parse(max\_x1.Text) },

{float.Parse(min\_x2.Text) ,float.Parse(max\_x2.Text) },};

if (Aa.Chom == 4)

Aa.interM = new float[,] {

{float.Parse(min\_x1.Text) ,float.Parse(max\_x1.Text) },

{float.Parse(min\_x2.Text) ,float.Parse(max\_x2.Text) },

{float.Parse(min\_x3.Text) ,float.Parse(max\_x3.Text) },

{float.Parse(min\_x4.Text) ,float.Parse(max\_x4.Text) }, };

// точки Следить что бы р1 был меньше р2

Aa.Ipoint = checkBox1.Checked;

Aa.point0 = float.Parse(VerPoint.Text);

if (float.Parse(VerPoint.Text) == 0.5) Aa.Ipoint = false;

if (Aa.Ipoint)

{

if (float.Parse(VerPoint.Text) < 0.5)

{

Aa.point1 =(int)Math.Floor(Aa.Length \* float.Parse(VerPoint.Text)); // точка скрещивания

Aa.point2 =(int)Math.Floor(Aa.Length \* (1 - float.Parse(VerPoint.Text)));

}

else

{

Aa.point2 =(int)Math.Floor(Aa.Length \* float.Parse(VerPoint.Text)); // точка скрещивания

Aa.point1 =(int)Math.Floor(Aa.Length \* (1 - float.Parse(VerPoint.Text)));

}

}

else

{

Aa.point1 = (int)Math.Floor(Aa.Length \* float.Parse(VerPoint.Text)); // точка скрещивания

Aa.point2 = 0;

}

/\*

if (checkBox2.Checked)

{

MtextBox.Text = MtextBox.Text + Aa.Chom + Environment.NewLine

+ Aa.interM[0, 0] + Aa.interM[0, 1] + Aa.interM[1, 0] + Aa.interM[1, 1] + Environment.NewLine

+ Aa.interM[2, 0] + Aa.interM[2, 1] + Aa.interM[3, 0] + Aa.interM[3, 1];

}

\*/

Aa.Kod = checkBox3.Checked;

Aa.Set=checkBox4.Checked;

Aa.DopIter=Convert.ToInt32(Dopit.Text);

Aa.Setka = Convert.ToInt32(Setka.Text);

}

private void checkBox2\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox2.Checked)

{

Aa.Chom = 4;

min\_x3.Enabled = true;

max\_x3.Enabled = true;

min\_x4.Enabled = true;

max\_x4.Enabled = true;

}

else

{

Aa.Chom = 2;

min\_x3.Enabled = false;

max\_x3.Enabled = false;

min\_x4.Enabled = false;

max\_x4.Enabled = false;

}

help\_box();

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

Aa.Ipoint = checkBox1.Checked;

help\_box();

}

private void checkBox3\_CheckedChanged(object sender, EventArgs e)

{

Aa.Kod = checkBox3.Checked;

help\_box();

}

private void button2\_Click(object sender, EventArgs e)

{

MtextBox.Text = "";

//Set\_param();

//Test\_Individ();

}

private void help\_box()

{

if (Aa.Kod)

Help\_textBox.Text = "Вещественная кодировка";

else

Help\_textBox.Text = "Целочисленная кодировка";

if (Aa.Chom == 2)

Help\_textBox.Text = Help\_textBox.Text + Environment.NewLine + "Две переменных (хром)";

else

Help\_textBox.Text = Help\_textBox.Text + Environment.NewLine + "Четыре переменных (хром)";

if (Aa.Ipoint)

Help\_textBox.Text = Help\_textBox.Text + Environment.NewLine + "Две точки скрещивания";

else

Help\_textBox.Text = Help\_textBox.Text + Environment.NewLine + "Одна точка скрещивания";

Help\_textBox.Text = Help\_textBox.Text + Environment.NewLine + Environment.NewLine + Aa.form;

}

}