**Лабораторная работа №7**

1. “Женихи и невесты". Есть N женихов и M невест. Каждый хочет найти свою пару.

У каждого есть свои предпочтения.

 Девушка хочет найти жениха. Кандидатов достаточно много. Известен список

бинарных свойств, которые девушка хочет знать о своих женихах. Часть из этих

свойств девушка относит к положительным, часть - к отрицательным. Девушка

хочет из списка кандидатов составить список предпочтительных кандидатов,

упорядоченный по степени предпочтения. Предложите ей разумный алгоритм и

реализуйте его.

Указание: список свойств следует задать перечислением, представляющим шкалу.

Следует определить класс Groom, среди полей которого будет поле properties,

заданное перечислением.

 Юноша хочет найти невесту. Кандидатов достаточно много. Известен список

бинарных свойств, которые молодой человек хочет знать о претендентках. Часть из

этих свойств юноша относит к положительным, часть - к отрицательным. Юноше

требуется из списка претенденток составить список предпочтительных невест,

упорядоченный по степени предпочтения. Предложите разумный алгоритм и

реализуйте его.

Указание: список свойств следует задать перечислением, представляющим шкалу.

Следует определить класс Bride, среди полей которого будет поле properties,

заданное перечислением.

Необходимо создать Windows-проект, моделирующий решение задачи создания пар. Эту

задачу можно рассматривать как вариацию известной задачи "об устойчивом

бракосочетании".

public partial class MarriageForm : Form

{

public MarriageForm()

{

InitializeComponent();

int defaultArraySize = 5;

textBoxBridesQuantity.Text = defaultArraySize.ToString();

textBoxGroomsQuantity.Text = defaultArraySize.ToString();

// событие: изменение выбранного элемента листбокса

listBoxBrides.SelectedIndexChanged += listBoxBrides\_SelectedIndexChanged;

listBoxGrooms.SelectedIndexChanged += listBoxGrooms\_SelectedIndexChanged;

Restart\_Pressed(ref BridesArray, ref GroomsArray, rand); //первичное заполнение "нажатием" кнопки рестарт

}

private Random rand = new Random();

private Bride[] BridesArray;

private Groom[] GroomsArray;

//создаем массивы, заполняем листбоксы

private void Restart\_Pressed(ref Bride[] BridesArray, ref Groom[] GroomsArray, Random rand)

{

listBoxBrides.Items.Clear();

listBoxGrooms.Items.Clear();

BridesArray = new Bride[int.Parse(textBoxBridesQuantity.Text.ToString())];

GroomsArray = new Groom[int.Parse(textBoxGroomsQuantity.Text.ToString())];

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

{

BridesArray[i] = new Bride($"{i}", rand);

listBoxBrides.Items.Add(BridesArray[i].name);

}

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

{

GroomsArray[i] = new Groom($"{i}", rand);

listBoxGrooms.Items.Add(GroomsArray[i].name);

}

listBoxBrides.SelectedIndex = 0;

listBoxGrooms.SelectedIndex = 0;

}

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

{

bool flag; //флаг для обнуления индекса цикла

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

{

flag = false;

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

{

// (если у невесты не мужа или если новый избранник лучже прежнего) И

// (у жениха нет неыесты или новая избранница лучше прежней)

if ((BridesArray[i].husband == null ||

BridesArray[i].EvaluateGroom(GroomsArray[j]) > BridesArray[i].EvaluateGroom(BridesArray[i].husband)) &&

(GroomsArray[j].wife == null ||

GroomsArray[j].EvaluateBride(BridesArray[i]) > GroomsArray[j].EvaluateBride(GroomsArray[j].wife)))

{

//собщаем бывшему мужу, что он в разводе

if (BridesArray[i].husband != null)

{

BridesArray[i].husband.wife = null;

flag = true;

}

BridesArray[i].husband = GroomsArray[j];

//собщаем бывшей жене, что она разведена

if (GroomsArray[j].wife != null)

{

GroomsArray[j].wife.husband = null;

flag = true;

}

GroomsArray[j].wife = BridesArray[i];

}

}

if (flag) i = -1;

}

// заполнение листбоксов попарно

listBoxBrides.Items.Clear();

listBoxGrooms.Items.Clear();

if (BridesArray.Length <= GroomsArray.Length)

{

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

{

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

{

if (BridesArray[i].husband == GroomsArray[j])

{

Groom tmp = GroomsArray[j];

GroomsArray[j] = GroomsArray[i];

GroomsArray[i] = tmp;

}

}

}

}

else

{

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

{

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

{

if (GroomsArray[i].wife == BridesArray[j])

{

Bride tmp = BridesArray[j];

BridesArray[j] = BridesArray[i];

BridesArray[i] = tmp;

}

}

}

}

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

{

listBoxBrides.Items.Add(BridesArray[i].name);

}

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

{

listBoxGrooms.Items.Add(GroomsArray[i].name);

}

listBoxBrides.SelectedIndex = 0;

listBoxGrooms.SelectedIndex = 0;

}

private void listBoxBrides\_SelectedIndexChanged(object sender, EventArgs e)

{

textBoxValues\_Fill();

}

private void listBoxGrooms\_SelectedIndexChanged(object sender, EventArgs e)

{

textBoxValues\_Fill();

}

private void textBoxValues\_Fill()

{

int brideIndex = listBoxBrides.SelectedIndex;

int groomIndex = listBoxGrooms.SelectedIndex;

if (brideIndex >= 0 && groomIndex >= 0)

{

textBoxValues.Text = $"{BridesArray[brideIndex].name}:";

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += BridesArray[brideIndex].traits;

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += $"{GroomsArray[groomIndex].name}:";

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += GroomsArray[groomIndex].traits;

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += Environment.NewLine;

int brideEvaluates = BridesArray[brideIndex].EvaluateGroom(GroomsArray[groomIndex]);

int groomEvaluates = GroomsArray[groomIndex].EvaluateBride(BridesArray[brideIndex]);

textBoxValues.Text += $"\"{BridesArray[brideIndex].name}\" likes \"{GroomsArray[groomIndex].name}\" for {brideEvaluates}";

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += Environment.NewLine;

textBoxValues.Text += $"\"{GroomsArray[groomIndex].name}\" likes \"{BridesArray[brideIndex].name}\" for {groomEvaluates}";

}

}

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

{

Restart\_Pressed(ref BridesArray, ref GroomsArray, rand);

}

}

[Flags]

enum Properities

{

Любезный = 1,

Обаятельный = 2,

Верный = 4,

Прилежный = 8,

Достойный = 16,

Грубый = 32,

Ленивый = 64,

Самолюбивый = 128

}

class Bride

{

public Bride(string name, Random rand)

{

this.name = "girl\_" + name;

for (int i = 0; i < 8; ++i)

{

traits |= (Properities)(Math.Pow(2, i) \* rand.Next(2));

if (i < 5)

desires[i] = rand.Next(1, 100);

else

desires[i] = -rand.Next(1, 100);

}

}

public int EvaluateGroom(Groom guy)

{

int value = 0;

Console.WriteLine($"\n{guy.name} is:");

for (int i = 0; i < 8; ++i)

{

Properities currentTrait = guy.traits & (Properities)(Math.Pow(2, i));

if (currentTrait > 0)

{

value += desires[i];

}

}

return value;

}

public Properities traits { get; }

public int[] desires = new int[8];

public string name { get; }

public Groom husband = null;

}

class Groom

{

public Groom(string name, Random rand)

{

this.name = "guy\_" + name;

for (int i = 0; i < 8; ++i)

{

traits |= (Properities)(Math.Pow(2, i) \* rand.Next(2));

if (i < 5)

desires[i] = rand.Next(1, 100);

else

desires[i] = -rand.Next(1, 100);

}

}

public int EvaluateBride(Bride girl)

{

int value = 0;

Console.WriteLine($"\n{girl.name} is:");

for (int i = 0; i < 8; ++i)

{

Properities currentTrait = girl.traits & (Properities)(Math.Pow(2, i));

if (currentTrait > 0)

{

value += desires[i];

}

}

return value;

}

public Properities traits { get; }

public int[] desires = new int[8];

public string name { get; }

public Bride wife = null;

}

}

2. Разработать динамическую библиотеку, содержащую класс вычисления функций с

помощью разложения в ряд Тейлора на интервале от x1 до x2 с шагом dx с

точностью eps. Разработать Windows-приложение для расчета требуемых значений. Пользователь

вводит в поля на форме значения x1, x2, dx, eps. Результат сравнить с выполнением

аналогичной функции класса Math.

По нажатии кнопки «Рассчитать» формируется таблица значений. Таблицу

снабдить заголовком и шапкой. Каждая строка таблицы должна содержать

значение аргумента, значение функции, значение функции класса Math, разница

двух значений и количество просуммированных членов ряда. Оценить время

работы программы.



Библиотека классов

namespace MathTools

{

public class Class1

{

public Class1(double x)

{

this.x = x;

EPS = 1e-6;

}

public double NewExpMinusX(double x)

{

double Sum = 1;

double An = Sum;

for (int i = 1; Math.Abs(An) > EPS; ++i)

{

An \*= -x / i;

Sum += An;

}

return Sum;

}

public double StandartExpMinusX(double x)

{

return Math.Exp(-x);

}

public double x;

public double EPS;

}

}

public partial class Form1:Form

{

public Form1()

{

Exponent = new MathTools.Class1(0.0);

InitializeComponent();

textBox\_inputx.Text = "0,0";

textBox\_inputeps.Text = $"{Exponent.EPS}";

\_n = 1;

textBox\_inputn.Text = $"{\_n}";

button\_Click1.Select();

}

private void Button1\_Click(object sender, EventArgs e)

{

Exponent.x = double.Parse(textBox\_inputx.Text);

Exponent.EPS = double.Parse(textBox\_inputeps.Text);

n = int.Parse(textBox\_inputn.Text);

textBox\_inputnew.Text = $"{Exponent.NewExpMinusX(Exponent.x)}";

label\_Time1.Text = $"ExecutionTime:" + $"{TimeValue.EvalTimeDToD(n, Exponent.NewExpMinusX, Exponent.x)}ms";

textBox\_inputmath.Text = $"{Exponent.StandartExpMinusX(Exponent.x)}";

label\_Time2.Text = $"ExecutionTime:" + $"{TimeValue.EvalTimeDToD(n, Exponent.StandartExpMinusX, Exponent.x)}ms";

}

MathTools.Class1 Exponent;

int \_n;

public int n

{

get

{

return \_n;

}

set

{

\_n =value;

}

}

}

public class TimeValue

{

public delegate double DToD(double arg1);

public static double EvalTimeDToD(int count,DToD fun,double x)

{

DateTime start,finish;

double res =0;

start =DateTime.Now;

for (int i=1;i<count;i++)

fun(x);

finish =DateTime.Now;

//res=(finish-start).Ticks;

res =(finish-start).Milliseconds;

return res;

}

}

}

Изображение выглядит как снимок экрана

Автоматически созданное описание

**Лабораторная работа №8**

1. Даны числа n и m. Создайте массив int A[n, m] и заполните следующим образом

(пример для n=5, m=6):

0 0 0 0 0 0

0 1 2 3 4 5

0 2 4 6 8 10

0 3 6 9 12 15

0 4 8 12 16 20

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

int m = Convert.ToInt32(textBox1.Text);

int n = Convert.ToInt32(textBox2.Text);

int[,] A = new int[m, n];

dataGridView1.RowCount = m;

dataGridView1.ColumnCount = n;

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

{

for (int j = 0; j < n; j++)

{

A[i, j] = i \* j;

dataGridView1.Rows[i].Cells[j].Value = A[i, j];

}

}

}

Изображение выглядит как снимок экрана

Автоматически созданное описание