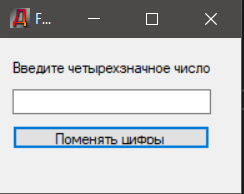
Вариант 6

**Задание 1**

Дано четырехзначное число. Поменяйте местами наименьшую и наибольшую цифры.

Внешний вид:



Код программы:

using System;

using System.Windows.Forms;

namespace \_1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

int n = int.Parse(textBox1.Text), min = 10, max = 0, min\_i = -1, max\_i = -1;

int[] m = new int[4];

for(int i = 0; i < 4; i++)

{

m[i] = GetDigit(n, i+1);

if (m[i] > max) { max = m[i]; max\_i = i; }

if (m[i] < min) { min = m[i]; min\_i = i; }

}

m[min\_i] = max;

m[max\_i] = min;

textBox1.Text = "";

for (int i = 0; i < 4; i++)

textBox1.Text += m[i];

}

private int GetDigit(int x, int digitNumber)

{

int digitCount = (int)Math.Log10(x) + 1;

if (digitNumber > digitCount)

return 0;

var pow = (int)Math.Pow(10, digitCount - digitNumber);

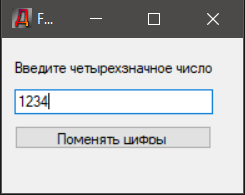
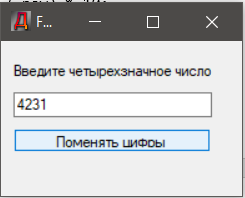
return (x / pow) % 10;

}

}

}

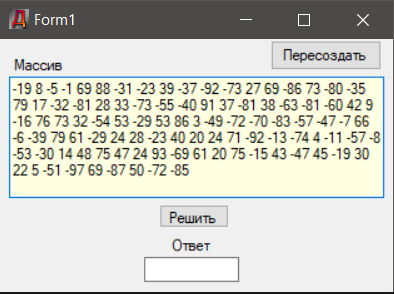
Тестирование:

**Задание 2**

Найдите сумму чисел массива, которые стоят на нечетных местах и при этом превосходят сумму крайних элементов массива.

Внешний вид:



Код программы:

using System;

using System.Windows.Forms;

namespace \_2

{

public partial class Form1 : Form

{

static int MAX\_ELEMS = 100;

int[] m = new int[MAX\_ELEMS];

public Form1()

{

InitializeComponent();

createArray();

}

private void createArray()

{

Random rnd = new Random();

textBox2.Text = "";

for (int i = 0; i < MAX\_ELEMS; i++)

{

m[i] = rnd.Next(-100, 100);

textBox2.Text += m[i] + " ";

}

}

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

{

createArray();

}

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

{

int s = m[0] + m[MAX\_ELEMS - 1];

int r = 0;

for (int i = 0; i < MAX\_ELEMS; i++)

{

if (i % 2 != 0 && m[i] > s) r += m[i];

}

textBox1.Text = "";

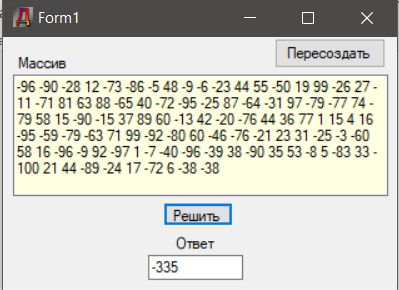
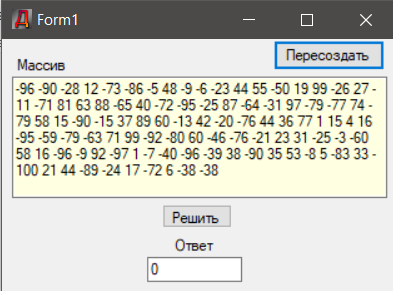
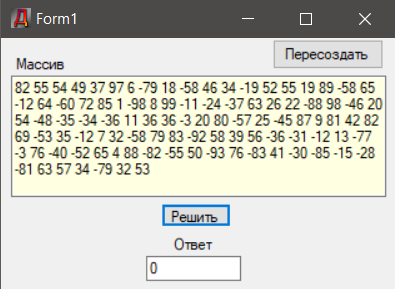
textBox1.Text = r.ToString();

}

}

}

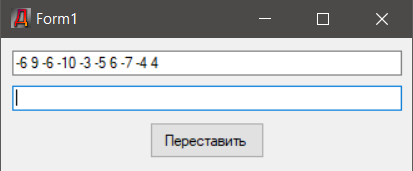
Тестирование:



**Задание 3**

Переставить элементы массива в обратном порядке.

Внешний вид:



Код программы:

using System;

using System.Windows.Forms;

namespace \_3

{

public partial class Form1 : Form

{

int[] m = new int[10];

public Form1()

{

InitializeComponent();

Random rnd = new Random();

for(int i = 0; i < 10; i++)

{

m[i] = rnd.Next(-10, 10);

textBox1.Text += m[i] + " ";

}

}

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

{

int[] buf = new int[5];

for (int i = 0; i < 5; i++)

{

buf[i] = m[i];

m[i] = m[9 - i];

}

for (int i = 0; i < 10; i++)

{

if (i < 5) m[9-i] = buf[i];

textBox2.Text += m[i] + " ";

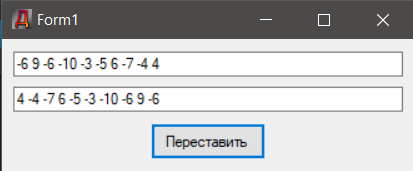
}

}

}

}

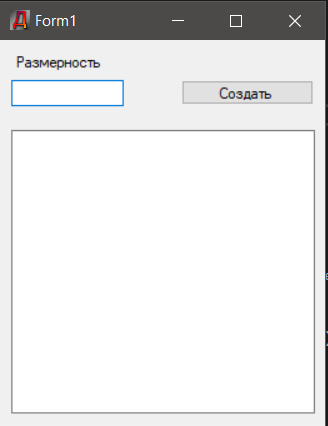
Тестирование:



**Задание 4**

Создать квадратную матрицу, на диагонали которой находятся тройки, выше диагонали находятся двойки, остальные элементы равна единице.

Внешний вид:



Код программы:

using System;

using System.Windows.Forms;

namespace \_4

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

textBox1.Text = "";

int size = int.Parse(textBox2.Text);

int[,] m = new int[size, size];

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

{

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

{

if (i == j) m[i, j] = 3;

if (j > i) m[i, j] = 2;

if (j < i) m[i, j] = 1;

textBox1.Text += m[i, j] + " ";

}

textBox1.Text += Environment.NewLine;

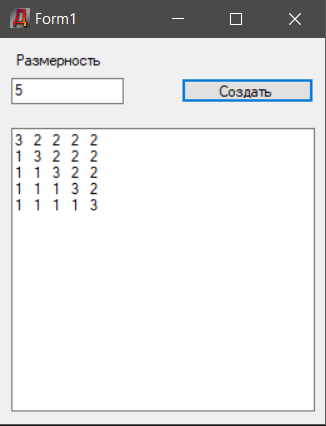
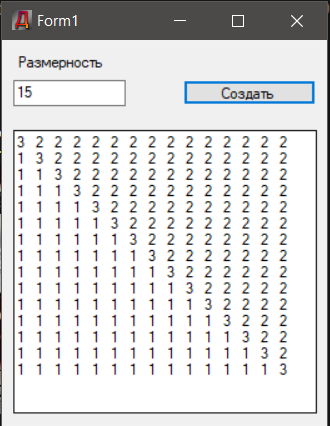
}

}

}

}

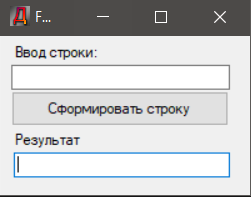
Тестирование:

**Задание 5**

Удалить в строке все цифры.

Внешний вид:



Код программы:

using System;

using System.Text.RegularExpressions;

using System.Windows.Forms;

namespace \_5

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

string s = textBox1.Text;

s = Regex.Replace(s, "[0-9]", "");

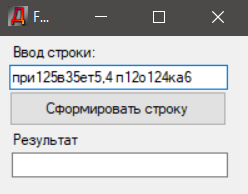
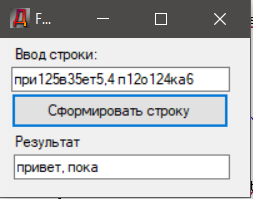
textBox2.Text = s;

}

}

}

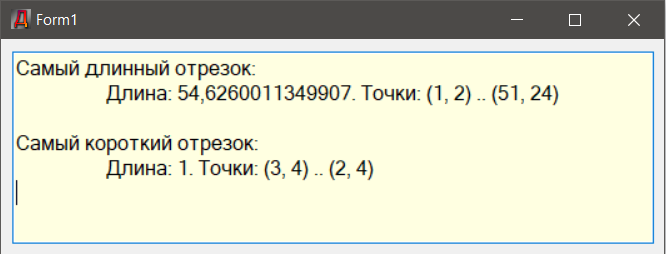
Тестирование:

**Задание 6**

В текстовый файл занесены пары чисел, разделенных пробелом (каждая пара чисел – в новой строке). Рассматривая каждую пару как координаты точек на плоскости, найти наибольшее и наименьшее расстояния между этими точками.

Внешний вид:



Код программы:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace \_6

{

public partial class Form1 : Form

{

List<Point> points = new List<Point>();

List<Line> lines = new List<Line>();

public Form1()

{

InitializeComponent();

string line;

System.IO.StreamReader file = new System.IO.StreamReader(@"coords.txt");

while ((line = file.ReadLine()) != null)

{

if (line.Length <= 0) continue;

string[] s = line.Split(' ');

Point p = new Point(double.Parse(s[0]), double.Parse(s[1]));

points.Add(p);

}

file.Close();

for(int i = 0; i < points.Count(); i++)

{

Point p = points[i];

for (int j = 0; j < points.Count(); j++)

{

Point pp = points[j];

if (p.X == pp.X && p.Y == pp.Y) continue;

bool flag = false;

Line temp = new Line(p, pp);

for (int l = 0; l < lines.Count(); l++)

{ if (temp.Len == lines[l].Len) { flag = true; break; } }

if (!flag)

lines.Add(temp);

}

}

int max\_i = 0, min\_i = 0;

for (int i = 0; i < lines.Count(); i++)

{

double cur = lines[i].Len;

if (cur > lines[max\_i].Len) max\_i = i;

if (cur < lines[min\_i].Len) min\_i = i;

}

textBox1.Text += "Самый длинный отрезок:"+Environment.NewLine;

textBox1.Text += string.Format("\tДлина: {0}. Точки: ({1}, {2}) .. ({3}, {4})", lines[max\_i].Len, lines[max\_i].P1.X, lines[max\_i].P1.Y, lines[max\_i].P2.X, lines[max\_i].P2.Y) +Environment.NewLine + Environment.NewLine;

textBox1.Text += "Самый коткий отрезок:" + Environment.NewLine;

textBox1.Text += string.Format("\tДлина: {0}. Точки: ({1}, {2}) .. ({3}, {4})", lines[min\_i].Len, lines[min\_i].P1.X, lines[min\_i].P1.Y, lines[min\_i].P2.X, lines[min\_i].P2.Y) +Environment.NewLine;

}

}

}

public struct Point

{

public Point(double x, double y)

{

X = x;

Y = y;

}

public double X { get; }

public double Y { get; }

}

public struct Line

{

public Line(Point p1, Point p2)

{

P1 = p1; P2 = p2;

Len = Math.Sqrt(Math.Pow((p2.X - p1.X), 2) + Math.Pow((p2.Y - p1.Y), 2));

}

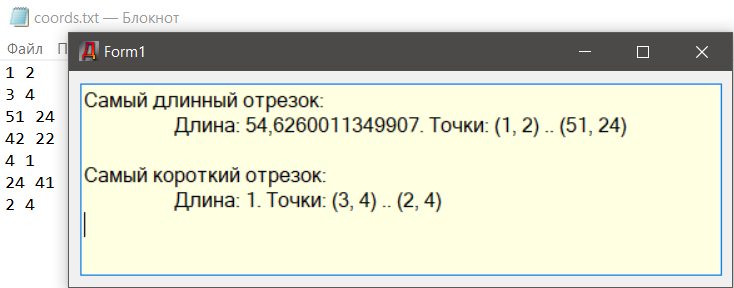
public double Len;

public Point P1 { get; }

public Point P2 { get; }

}

Тестирование:



**Задание 7**

1. Описать структуру с именем WORKER, содержащую следующие поля:

• NAME — фамилия и инициалы работника;

• POS — название занимаемой должности;

• YEAR — год поступления на работу.

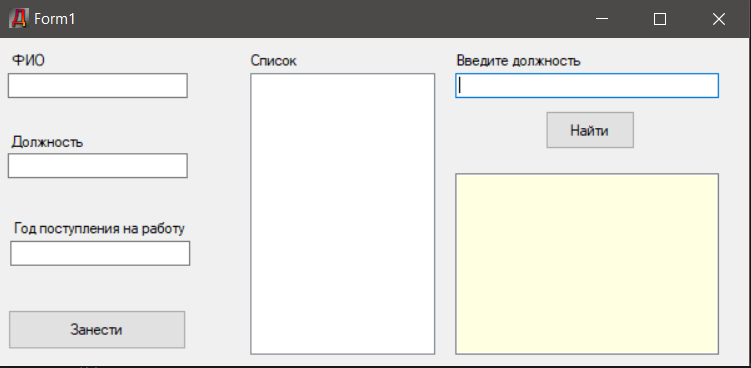
2. Написать программу, выполняющую следующие действия:

• ввод с клавиатуры данных в массив TABL, состоящий из десяти структур типа WORKER; записи должны быть размещены по алфавиту.

• вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры;

• если таких работников нет, вывести на дисплей соответствующее сообщение.

Внешний вид:



Код программы:

using System;

using System.Windows.Forms;

namespace \_7

{

public partial class Form1 : Form

{

WORKER[] list = new WORKER[10];

int counter = 0;

public Form1()

{

InitializeComponent();

}

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

{

if(textBox1.TextLength > 0 && textBox2.TextLength > 0 && textBox3.TextLength > 0 && counter < 10)

{

list[counter] = new WORKER(textBox1.Text, textBox2.Text, textBox3.Text);

listBox1.Items.Add(list[counter].NAME + " - " + list[counter].POST);

counter++;

textBox1.Text = textBox2.Text = textBox3.Text = "";

}

}

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

{

if (textBox4.TextLength > 0)

{

textBox5.Text = "";

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

{

if (list[i].POST.ToLower() == textBox4.Text.ToLower())

textBox5.Text += list[i].NAME+", "+list[i].POST+", "+list[i].YEAR + Environment.NewLine;

}

if (textBox5.TextLength == 0) textBox5.Text = "Нет сотрудников с такой должностью";

}

}

}

}

public struct WORKER

{

public string NAME;

public string POST;

public string YEAR;

public WORKER(string name, string post, string year)

{

NAME = name;

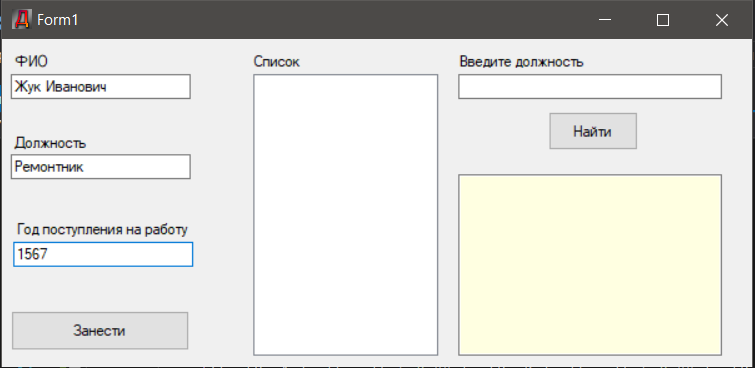
POST = post;

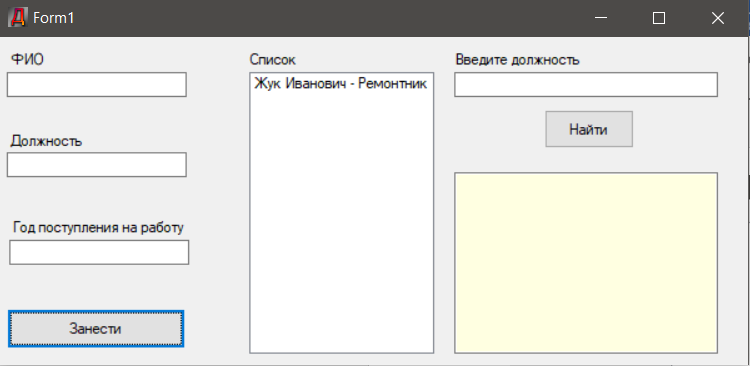
YEAR = year;

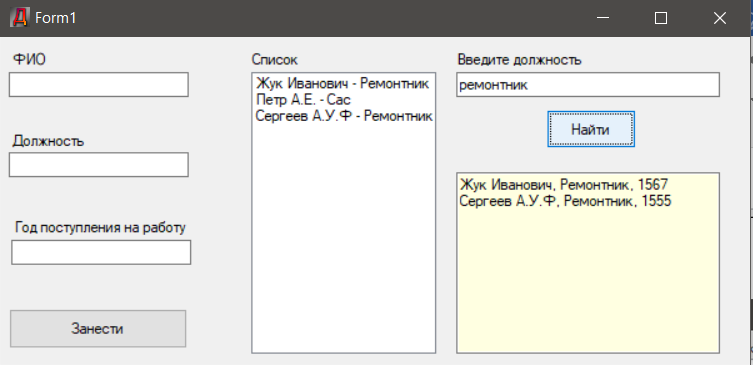
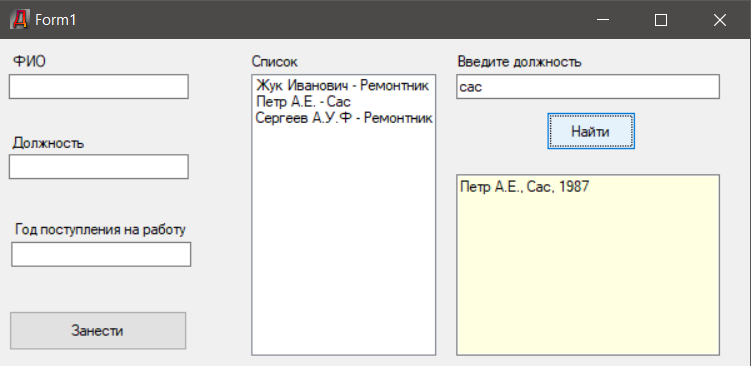
}

}

Тестирование:





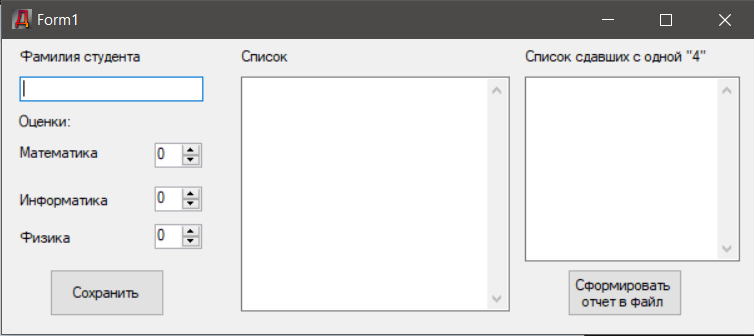
 

**Задание 8**

Имеется информация по итогам экзаменов в институте всего в списке N человек. По каждому из студентов имеются следующие сведения: фамилия, оценка по математике, оценка по информатике и оценка по физике. Ввести информацию об экзаменах и напечатать количество и фамилии студентов, которые сдали экзамены только с одной четверкой.

Результаты сохранить в файл.

Внешний вид:



Код программы:

using System;

using System.IO;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_8

{

public partial class Form1 : Form

{

List<STUDENT> list = new List<STUDENT>();

int kol = 0;

public Form1()

{

InitializeComponent();

}

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

{

if (textBox1.TextLength > 0 && numericUpDown1.Value <= 5 && numericUpDown2.Value <= 5 && numericUpDown3.Value <= 5)

{

list.Add(new STUDENT(textBox1.Text, (int)numericUpDown1.Value, (int)numericUpDown2.Value, (int)numericUpDown3.Value));

label6.Text = "Добавлен "+textBox1.Text;

textBox1.Text = "";

STUDENT t = list[list.Count() - 1];

if (

(t.M == 4 && t.I !=4 && t.P!=4) ||

(t.I == 4 && t.P != 4 && t.M != 4) ||

(t.P == 4 && t.M != 4 && t.I != 4)

)

{

label8.Text= "Список сдавших с одной \"4\" - " + (++kol);

textBox3.Text += t.NAME + Environment.NewLine;

}

updateList();

}

}

private void updateList()

{

textBox2.Text = "";

for(int i = 0; i < list.Count(); i++)

{

STUDENT t = list[i];

textBox2.Text += t.NAME +

Environment.NewLine + "Математика: " + t.M +

Environment.NewLine + "Информатика: " + t.I +

Environment.NewLine + "Физика: " + t.P +

Environment.NewLine + "------------------" + Environment.NewLine;

}

}

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

{

using (StreamWriter sw = new StreamWriter("report.txt", false, System.Text.Encoding.UTF8))

{

sw.WriteLine("["+kol+"]"+ Environment.NewLine+textBox3.Text);

}

MessageBox.Show("Отчет сохранен в report.txt");

}

}

}

public struct STUDENT

{

public string NAME;

public int M, I, P;

public STUDENT(string name, int m, int i, int p)

{

NAME = name;

I = i;

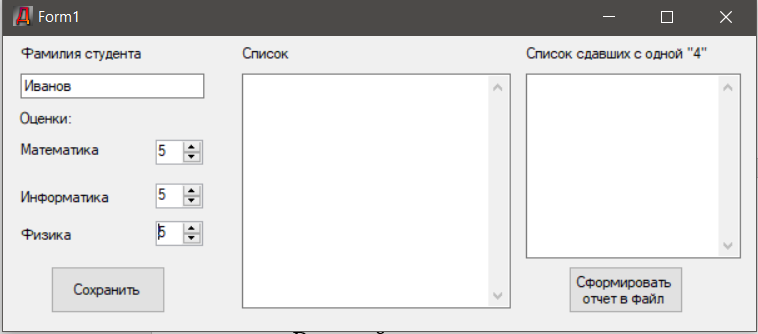
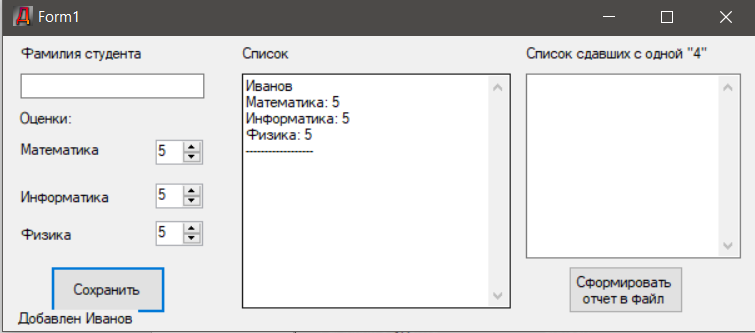
M = m;

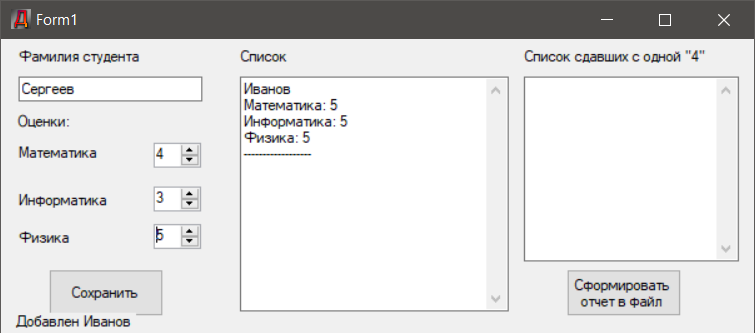
P = p;

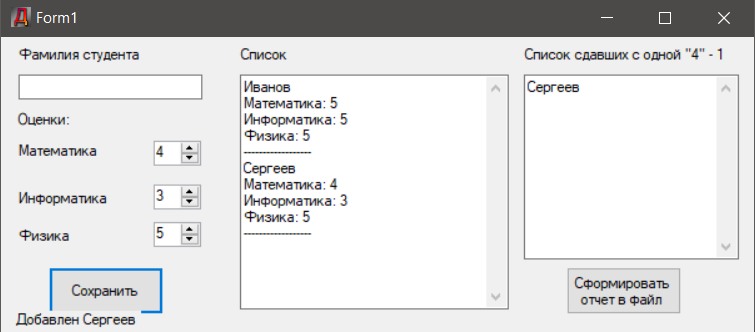
}

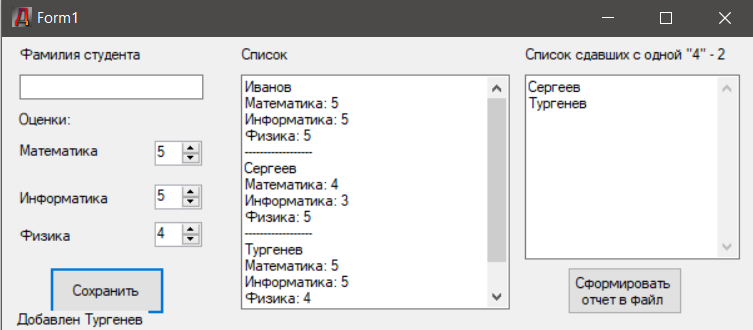
}

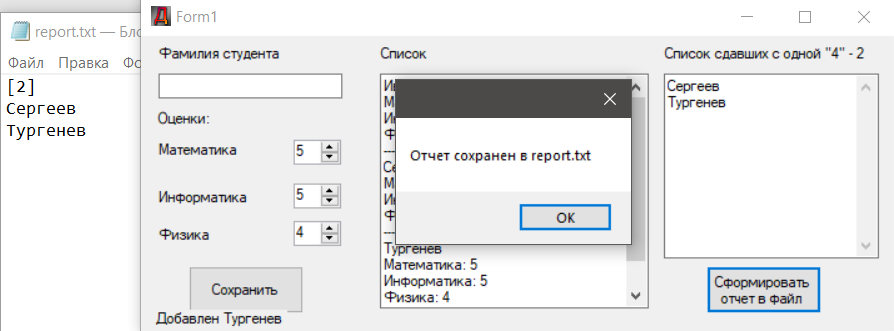
Тестирование:



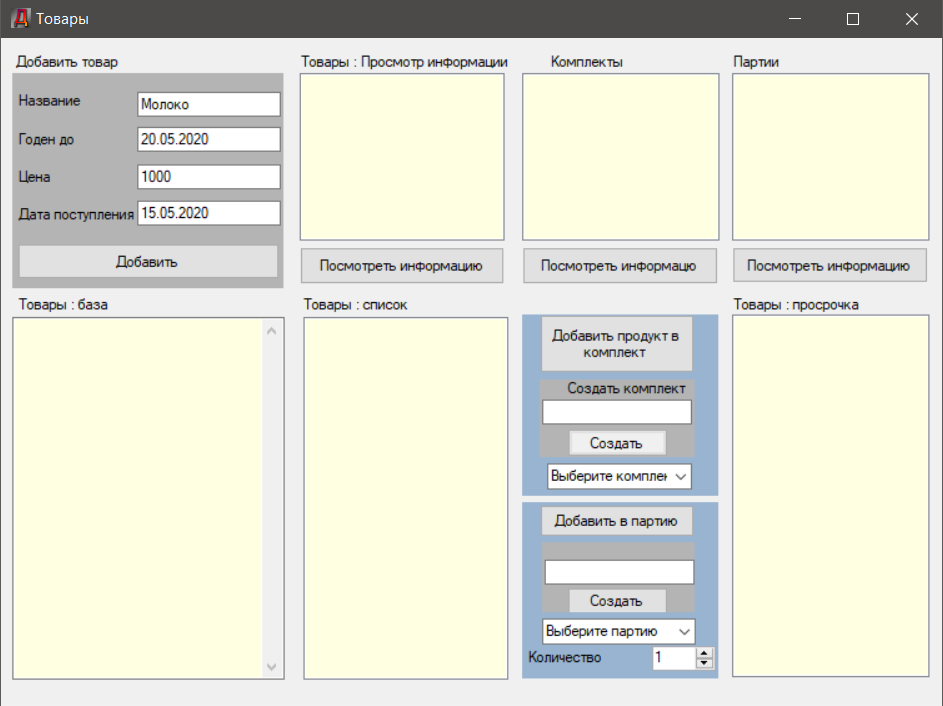






**Задание 9**

Создать абстрактный класс «Товар» с методами, позволяющими вывести на экран информацию о товаре, а также определить, соответствует ли она сроку годности на текущую дату. Создать производные классы: Продукт (название, цена, дата производства, срок годности), Партия (название, цена, количество шт, дата производства, срок годности), Комплект (названия, цена, перечень продуктов) со своими методами вывода информации на экран, и определения соответствия сроку годности. Создать базу (массив) из n товаров, вывести полную информацию из базы на экран, а также организовать поиск просроченного товара (на момент текущей даты).

Внешний вид: 

Код программы:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace \_9

{

public partial class Form1 : Form

{

List<Product> plist = new List<Product>();

List<Kit> kitlist = new List<Kit>();

List<Party> partylist = new List<Party>();

List<string> kitnamelist = new List<string>();

List<string> partynamelist = new List<string>();

int selectedAll = -1, selectedList = -1, selectedParty = -1;

public Form1()

{

InitializeComponent();

ntb.Text = "Молоко";

stb.Text = "20.05.2020";

ptb.Text = "1000";

dtb.Text = "15.05.2020";

}

/\*

\* Converting string date like "01.02.2020" to UNIX time if dir == true

\* Converting UNIX time to string if dir == false

\*/

private string str\_date(string t, bool dir)

{

if (dir)

{

string[] strdate = t.Split('.');

var dt = new DateTime(int.Parse(strdate[2]), int.Parse(strdate[1]), int.Parse(strdate[0]), 0, 0, 0, DateTimeKind.Local);

var epoch = new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

var unixDateTime = (dt - epoch).TotalSeconds;

return unixDateTime.ToString();

}

else

{

DateTime dtDateTime = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);

dtDateTime = dtDateTime.AddSeconds(int.Parse(t)).ToLocalTime();

string r = dtDateTime.Day.ToString() + "." + dtDateTime.Month.ToString() + "." + dtDateTime.Year.ToString();

return r;

}

}

private void updateLB(ListBox lb)

{

lb.Items.Clear();

switch (lb.Name.ToString())

{

case "listBox1":

for (int i = 0; i < plist.Count(); i++)

lb.Items.Add(plist[i].name);

break;

case "listBox4":

for (int i = 0; i < plist.Count(); i++)

lb.Items.Add(plist[i].name);

break;

case "listBox2":

for (int i = 0; i < kitlist.Count(); i++)

lb.Items.Add(kitlist[i].kitname);

break;

case "listBox3":

for (int i = 0; i < partylist.Count(); i++)

lb.Items.Add(partylist[i].pname);

break;

default:

MessageBox.Show("Ошибка определения листбокса");

break;

}

}

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

{

if (ntb.TextLength > 0 && ptb.TextLength > 0 && stb.TextLength > 0 && dtb.TextLength > 0)

{

try

{

var p = new Product(ntb.Text, int.Parse(str\_date(stb.Text, true)), int.Parse(ptb.Text), int.Parse(str\_date(dtb.Text, true)));

//MessageBox.Show(p.getSrok());

plist.Add(p);

updateLB(listBox1);

updateLB(listBox4);

updateOutOfDate();

textBox2.Text = "";

for (int i = 0; i < plist.Count(); i++)

textBox2.Text += ("================"

+ Environment.NewLine

+ "Название: "+plist[i].name

+ Environment.NewLine

+ "Годен до: "+ str\_date(plist[i].srok.ToString(), false)

+ Environment.NewLine

+ "Цена: " + plist[i].price

+ Environment.NewLine

+ "Дата производства: " + str\_date(plist[i].date.ToString(), false)

+ Environment.NewLine);

ntb.Text = ptb.Text = stb.Text = dtb.Text = "";

}

catch

{

MessageBox.Show("Проверьте правильность введенных данных", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

public void updateOutOfDate()

{

listBox5.Items.Clear();

for (int i = 0; i < plist.Count(); i++)

{

int curtime = (int)(DateTime.UtcNow - new DateTime(1970, 1, 1)).TotalSeconds;

if (plist[i].srok - curtime < 0)

listBox5.Items.Add(plist[i].name);

}

}

private void listBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

if(listBox2.SelectedIndex != -1)

selectedList = listBox2.SelectedIndex;

}

private void button5\_Click(object sender, EventArgs e)

{

if (selectedList >= 0)

{

kitlist[selectedList].PrintKit();

}

}

private void button3\_Click(object sender, EventArgs e)

{

if(listBox4.SelectedIndex >= 0 && listBox4.SelectedIndex >= 0 && comboBox1.SelectedIndex >= 0)

{

bool flag = false;

bool flag1 = false;

int \_i = 0;

for (int i = 0; i < kitlist.Count(); i++)

{

if(kitlist[i].kitname == comboBox1.SelectedItem.ToString())

{

flag = true;

\_i = i;

}

if(kitlist[i].kitname == listBox4.SelectedItem.ToString())

{

flag1 = true;

}

}

if (!flag1)

{

if (flag)

{

kitlist[\_i].list.Add(plist[listBox4.SelectedIndex]);

MessageBox.Show("Добалвено!");

}

else

MessageBox.Show("Выберите название комплекта из списка ниже");

}

else

MessageBox.Show("Такой товар уже есть в комплекте");

}

else

MessageBox.Show("Выберите название комплекта из списка ниже");

}

private void button7\_Click(object sender, EventArgs e)

{

if (textBox1.TextLength > 0)

{

kitnamelist.Add(textBox1.Text);

kitlist.Add(new Kit(textBox1.Text));

comboBox1.Items.Clear();

listBox2.Items.Clear();

for (int i = 0; i < kitnamelist.Count(); i++)

{

comboBox1.Items.Add(kitnamelist[i]);

listBox2.Items.Add(kitnamelist[i]);

}

updateLB(listBox2);

textBox1.Text = "";

}

}

private void button8\_Click(object sender, EventArgs e)

{

if (textBox3.TextLength > 0)

{

partynamelist.Add(textBox3.Text);

partylist.Add(new Party(textBox3.Text, (int)numericUpDown1.Value));

comboBox2.Items.Clear();

listBox3.Items.Clear();

for (int i = 0; i < partynamelist.Count(); i++)

{

comboBox2.Items.Add(partynamelist[i]);

listBox3.Items.Add(partynamelist[i]);

}

}

updateLB(listBox3);

textBox3.Text = "";

}

private void button4\_Click(object sender, EventArgs e)

{

if(comboBox2.SelectedIndex >= 0 && listBox4.SelectedIndex >= 0)

{

bool flag = false;

bool flag1 = false;

int \_i = 0;

for (int i = 0; i < partylist.Count(); i++)

{

if (partylist[i].pname == comboBox2.SelectedItem.ToString())

{

flag = true;

\_i = i;

}

if (partylist[i].pname == listBox4.SelectedItem.ToString())

flag1 = true;

}

if (!flag1)

{

if (flag)

{

partylist[\_i].list.Add(plist[listBox4.SelectedIndex]);

MessageBox.Show("Добалвено!");

}

else

MessageBox.Show("Выберите название комплекта из списка ниже");

}

else

MessageBox.Show("Такой товар уже есть в партии");

}

else

MessageBox.Show("Такой товар уже есть в партии");

}

private void button6\_Click(object sender, EventArgs e)

{

if (selectedParty >= 0)

{

partylist[selectedParty].PrintParty();

}

}

private void listBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

selectedParty = listBox3.SelectedIndex;

}

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

if(listBox1.SelectedIndex != -1)

selectedAll = listBox1.SelectedIndex;

}

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

{

if(selectedAll >= 0)

{

string s = plist[selectedAll].getSrok();

if (s == "out") s = "Срок годности вышел";

MessageBox.Show(s);

}

}

}

}

abstract class Tovar

{

public string name;

public int srok, price, date;

public string getSrok()

{

int curtime = (int)(DateTime.UtcNow - new DateTime(1970, 1, 1)).TotalSeconds;

int rem = srok - curtime;

return (rem > 0) ? "Продукт годен еще " + TStoStr(rem): "out";

}

private string TStoStr(int ts)

{

string result = "";

double hours = ts / 3600, days = 0;

if(hours > 24)

{

days = Math.Floor(hours / 24);

hours -= (days \* 24);

}

result = (days == 0) ? hours + " часов" : days + " дня(дней) и "+ hours + " часов";

return result;

}

}

class Product : Tovar

{

private string inttostr(long t)

{

DateTime dtDateTime = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);

dtDateTime = dtDateTime.AddSeconds(t).ToLocalTime();

string r = dtDateTime.Day.ToString() + "." + dtDateTime.Month.ToString() + "." + dtDateTime.Year.ToString();

return r;

}

public Product(string n, int s, int p, int d)

{

name = n;

srok = s;

price = p;

date = d;

}

public string get(bool n = false)

{

return n

?string.Format("Название: {0}, Срок годности: {1}, Цена: {2}, Дата поступления: {3}", name, inttostr(srok), price, inttostr(date))

:string.Format("Название: {0}"+Environment.NewLine + "Срок годности: {1}"+Environment.NewLine + "Цена: {2}"+Environment.NewLine + "Дата поступления: {3}", name, inttostr(srok), price, inttostr(date));

}

}

class Party : Tovar

{

public List<Product> list = new List<Product>();

public string pname { get; }

private int kolvo;

public Party(string pn, int k)

{

kolvo = k;

pname = pn;

}

public void PrintParty()

{

string ret = "";

if (kolvo != -1)

ret = Environment.NewLine + "Количество: " + kolvo + Environment.NewLine;

string l = "=============\n";

int i;

for (i = 0; i < list.Count(); i++)

l += list[i].get() + ret + "=============\n";

if (i != 0)

MessageBox.Show("Партия \"" + pname + "\"\nТовары:\n" + l);

else

MessageBox.Show("Партия пустая");

}

}

class Kit : Tovar

{

public List<Product> list = new List<Product>();

public string kitname;

public Kit(string kn)

{

kitname = kn;

}

public void PrintKit()

{

string l = "=============\n";

int i;

for (i = 0; i < list.Count(); i++)

l += list[i].get() + "\n=============\n";

if (i != 0)

MessageBox.Show("Комплект \"" + kitname + "\"\nТовары:\n" + l);

else

MessageBox.Show("Комплект пустой");

}

}

Тестирование:

\*не смог разобраться, как unix время конвертировать правильно, поэтому

