«Национальный открытый институт»

Сибирев В.Н. Рачева Н.В.

РАЗРАБОТКА, ВНЕДРЕНИЕ И АДАПТАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ОТРАСЛЕВОЙ НАПРАВЛЕННОСТИ
(интегрированная среда C# Visual Studio.NET)

Методические указания
к выполнению практических занятий

Рекомендовано Методической комиссией по качеству
Национального открытого института
для студентов, обучающихся по направлению
09.03.03 «Прикладная информатика»

Санкт-Петербург
2016

УДК 519.2.06 (07)
ББК 32.973
С34

Методические указания разработаны на основе рабочей программы “Прикладная информатика (по отраслям)” в соответствии с требованиями государственных образовательных стандартов среднего профессионального образования.

Практические задания выполняются студентами самостоятельно по индивидуальному варианту. Методические указания предназначены для студентов специальность – 090205 Прикладная информатика (по отраслям)

УДК 519.2.06 (07)
ББК 32.973

1. Теоретические сведения

Программирование в среде VisualStudio.NET

Автором (один из создателей фундамента) языка программирования **C#** является *Андерс Хейлсберг*, кто проявил себя в проектах TurboPascal и Delphi. Язык **C#** - логическое продолжение **C++** и конкурент **Java,** особенность **C#** - это ориентация на технологии платформы **.NET.**

.NET*Framework* - интегрированная среда программирования, предложенная Microsoft для написания программных приложений, выполняющая совмещение всех программных средств и языков под управлением системы CLR (*CommonLanguageRuntime*) с промежуточной компиляцией на псевдокоде.

Практические задания предполагается выполнять в среде Visual C# Express – бесплатной версии интегрированной среды серии ExpressEdition.

Практическое задание 1

Цель задания: приобрести навыки по обработке однотипной информации, размещенной в виде массивов данных в оперативной памяти.

Задача:

1. ввести данные в оперативную память с клавиатуры;
2. разместить данные в виде двумерного массива;
3. написать программный код на **C#**;
4. проверить работоспособность программы.

Варианты для индивидуальной работы:

(вариант выбирается из двух пунктов по согласованию с преподавателем)

Размерность и расположение данных на листах **Excel**произвольно.

1. ***Выполнить*** выборку из двумерного массива:
   1.1 отрицательных значений;
   1.2 значений больше заданной константы, константа произвольная;
   1.3 значений, размещенных в чётных столбцах;

1.4 всех значений в заданном интервале, границы произвольные.

1.5 положительных значений;

1.6 значений меньше заданной константы, константа произвольная;

1.7 значений, размещенных в нечётных столбцах;

1.8 значений, размещенных на главной диагонали;

1.9 значений, размещенных выше главной диагонали;

1.10 значений, размещенных ниже главной диагонали.

2. ***Вычислить***

2.1 количество значений больше заданной константы;

2.2 среднеарифметическое значений из выборки по п.1.4;

2.3 количество отрицательных значений из выборки по п.1.4;

2.4 количество значений меньше заданной константы;

2.5 сумму положительных значений из выборки по п.1.4;

2.6 сумму отрицательных значений из выборки по п.1.4;

2.7 количество значений больше заданной константы из выборки по п.1.4;

2.8 среднеарифметическое всех значений массива;

2.9 количество отрицательных значений из всего массива;

2.10 сумму положительных значений всего массива.

Пример программного кода по практическому заданию 1:

Условие задачи:

Разместить числовые данные (ввод с клавиатуры) в оперативной памяти в виде двумерного массива и вычислить сумму.

Исходный текст программного модуля

```
usingSystem;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication4
{
class mass
    {
```

```
publicint m = 2,n=3;
public float[,] x=new float [3,2];
public float summa()
        {
int i,j;
float s=0;
for (i=0; i<n; i++)
for (j=0; j<m; j++)
             s+=x[i,j];
return(s);
        }
        //функцияввода
public void inmas()
        {
string str;
            int i,j;
for (i=0; i<n; i++)
             for (j = 0; j < m; j++)
             {
Console.Write("mas[" + i + "," + j + "]=");
                 str = Console.ReadLine();
                 x[i, j] = float.Parse(str);
}
  //float.Parse - перевод «символ - число»
}
        //главнаяфункцияMain
    }
class Program
    {
static void Main(string[] args)
        {
            mass z = new mass();
            z.inmas();
float sum = z.summa();
Console.WriteLine("  Сумма=" + sum);
Console.ReadKey();
        }
    }
}
```

Практическое задание 2

Цель работы: закрепить практические навыки в организации информационных файлов символьного типа на базе интегрированной среды VisualStudio.NET.

Задача:

Создать последовательный файл произвольной символьной информации, предусмотреть добавление данных и чтение содержимого файла. Написатьпрограммный код отдельными программными модулями и объединенным кодом с управлением через меню [4].

Пример программного кода по практическому заданию 2:

Создание текстового файла

```
usingSystem;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
namespace ConsoleApplication2
{
class In_rec
    {
static void Main()
        {
string str;
int k = 0;
FileStream fout;
try
            {

fout = new
FileStream("test.txt",FileMode.Create);
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
```

```
return;
            }
StreamWriter F_out = new StreamWriter(fout);
            //конецввода"stop"
Console.WriteLine
(" Enter text <stop to quit>");
do
            {
Console.Write("...:");
                str= Console.ReadLine();
if (str !="stop")
                {
                     str=str+"\r\n";
try
                    {
F_out.Write(str);
k++;
                    }
catch (IOException exc)
                    {
Console.WriteLine(exc.Message);
break;
                    }
                }
            } while (str !="stop");
Console.WriteLine(" K=" + k);
F_out.Close();
        }
    }
}
```

Добавление записей в существующий файл

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
namespace ConsoleApplication4
{
```

```
class In_rec
        {
static void Main()
            {
string str;
int k = 0;
StreamWriter f_in;
                try
                {
    f_in = new StreamWriter("test.txt");
                }
catch (IOException exc)
                {
Console.WriteLine(exc.Message);
return;
                }
do
                {
Console.Write("Текстиз:");
              str = Console.ReadLine();
              if (str != "stop")
                    {
              str = str + "\r\n";
try
                          {
              f_in.Write(str); k++;
                          }
catch (IOException exc)
                          {
        Console.WriteLine(exc.Message);
              break;
                          }
                    }
                } while (str != "stop");
Console.WriteLine(" Added K=" + k);
f_in.Close();
            }
        }
    }
```

Чтение содержимого текстового файла

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace ConsoleApplication5
{
class In_rec
    {
static void Main()
        {
string str;
int k = 0;
FileStream f_in;
try
            {
f_in = new FileStream("test.txt",
FileMode.Open);
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
return;
            }
StreamReader f = new StreamReader(f_in);
            //====================
Console.WriteLine(" Text of file ");
try
            {
while ((str = f.ReadLine()) != null)
                {
Console.Write(str); k++;
                }
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
```

```
                //          break;
}
            f.Close();
        }
    }
}
```

Работа с текстовыми файлами через меню

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace ConsoleApplication1
{
class In_Out
    {
public void Add_rec()
            {
string str;
int k = 0;
bool appendFlag = true;
StreamWriter f_add;
try
            {
                    f_add = new
StreamWriter("Data1.txt",appendFlag);
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
return;
            }
     //====================
Console.WriteLine
(" Enter added text <stop to quit>");
do
            {
```

```
Console.Write("...:");
                str = Console.ReadLine();
if (str != "stop")
                {
                    str = str + "\r\n";
                    try
                    {
           f_add.Write(str); k++;
                    }
catch (IOException exc)
                    {
Console.WriteLine(exc.Message);
break;
                    }
                }
            } while (str != "stop");
Console.WriteLine(" Added K=" + k);
            f_add.Close();
        }

    //================================
public void In_rec()
    {
string str;
int k = 0;
FileStream fout;
try
            {
    fout = new FileStream
("Data1.txt", FileMode.Create);
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
return;
            }
StreamWriter F_out = new
StreamWriter(fout);
            //==================
Console.WriteLine
```

```
(" Enter text <stop to quit>");
do
            {
Console.Write("...:");
                str = Console.ReadLine();
if (str != "stop")
                {
                    str = str + "\r\n";
                    try
                    {
                    F_out.Write(str); k++;
                    }
catch (IOException exc)
                    {
Console.WriteLine(exc.Message);
break;
                    }
                }
            } while (str != "stop");
Console.WriteLine(" K=" + k);
            F_out.Close();
        }
        //==================================

public void Out_rec()
    {
string str;
int k = 0;
FileStream f_in;
try
            {
       f_in = new FileStream("Data1.txt",
FileMode.Open);
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
return;
            }
StreamReader f = new
```

```
StreamReader(f_in);
            //==================
Console.WriteLine(" Text of file ");
try
            {
while ((str = f.ReadLine()) != null)
                {
Console.WriteLine(str); k++;
                }
            }
catch (IOException exc)
            {
Console.WriteLine(exc.Message);
                //          break;
}
Console.WriteLine
("   В файле К="+k+" записей");
Console.ReadKey();
            f.Close();
        }
    }
     class Program
    {
static void Main(string[] args)
        {
string ans;
          In_Out z = new In_Out();
while (true)
          {
Console.WriteLine
(" Созданиефайла - A");
Console.WriteLine(" Добавление записи - B");
Console.WriteLine(" Просмотр файла - C");
Console.WriteLine(" Выход из программы - D");
Console.Write(" .... Выборрежима:");
              ans = Console.ReadLine();
switch (ans)
              {
case "A": z.In_rec();
                      break;
```

```
case "B": z.Add_rec();
break;
case "C": z.Out_rec();
break;
case "D": return;
//default:Console.WriteLine
//             ("ВыборотАдоD");
}
          }
        }
    }
}
```

Практическое задание 3

Цель работы: приобрестипрактические навыки в реализации задач по частичной автоматизации обработки информации с использованием средств интегрированной среды **VBA**.

Задача:

Разработать форму пользователя и написатьпрограммный код на заданную тему по выбранному варианту.

Варианты для индивидуальной работы:

Вариант 1

Создать Windows-приложение "*Формирование программы строительно-монтажных работ*". Предусмотреть:

1. Создание входных форм:

Смета работ (Код объекта, Наименование объекта, Код и Объем работы);

Справочник расценок (Код работы, Наименование работы, Расценка)

2. На основе входных данных построить выходной документ:

Стоимость работ (Код объекта, Итоговая стоимость работ).

3. Выполнить расчет итоговой стоимости работ по заданному объекту.

Вариант 2

Создать Windows-приложение "*Формирование счетов на оплату в автосервисе*". Предусмотреть:

1. Создание входных форм:

Справочник видов работ (Код работы, Марка автомобиля, Наименование работы, Стоимость работы);

Заказы (Дата, Номер заказа, Клиент, Марка автомобиля, Код работы).

2. На основе входных данных построить выходной документ:

Счет (Код объекта, Итоговая стоимость работ).

3. Выполнить расчет стоимости заказа по заданному Номеру заказа.

Вариант 3

Создать Windows-приложение "*Учет оплаты заказов* ". Предусмотреть:

1. Создание входных форм:

Справочник услуг (Код услуги, Наименование услуги, Стоимость услуги);

Клиенты (Код клиента, Наименование клиента, Адрес, Телефон).

2. На основе входных данных построить выходной документ:

Заказы (Номер заказа, Дата заказа, Код клиента, Код услуги, Стоимость услуги, Форма оплаты).

3. Выполнить расчет суммы оплаты заказов по заданной форме оплаты.

Вариант 4

Создать Windows-приложение "*Учет торговых заказов* ". Предусмотреть:

1. Создание входных форм:

Каталог товаров (Код товара, Наименование товара, Цена товара);

Клиенты (Код клиента, Наименование клиента, Адрес, Телефон).

2. На основе входных данных построить выходной документ:

Заказы (Номер заказа, Дата заказа, Код клиента, Код товара, Количество).

3. Выполнить расчет стоимости заказа по заданному Номеру заказа.

Вариант 5

Создать Windows-приложение "*Формирование реестразаказов* ". Предусмотреть:

1. Создание входных форм:

Каталог услуг (Код услуги, Наименование услуги, Стоимость услуги);

Клиенты (Код клиента, ФИО клиента, Адрес, Телефон).

2. На основе входных данных построить выходной документ:

Заказы (Номер заказа, Дата заказа, Код клиента, Код услуги, Стоимость услуги).

3. Выполнить расчет общей стоимости заказа по заданному Номеру заказа.

Вариант 6

Создать Windows-приложение "*Учет движения продукциина складе* ". Предусмотреть:

1. Создание входных форм:

Движение (Код продукции, Наименование, Остаток на начало);

Накладные на поступление (Номер накладной, Код продукции, Количество поступлений):

Требования на выдачу (Номертребования, Код продукции, Количество выдано).

2. На основе входных данных построить выходной документ:

Учет остатков(Код продукции, Остаток на конец).

3. Выполнить расчет остатков на начало и конец срока по заданному коду продукции.

Вариант 7

Создать Windows-приложение " *Расчет зарплаты к выдаче работникам* "**.**

Предусмотреть:

1. Создание входных форм:

Справочник работников (Табельный номер, ФИО работника, Код отдела, Количество льгот);

Начисления (Табельный номер, начислено).

2. На основе входных данных построить выходной документ:

Удержания (Табельный номер, Удержания в ПФ, Удержания ПН).

3. Выполнить расчет зарплаты к выдаче по заданному табельному номеру.

Вариант 8

Расчет удержаний с зарплаты

Создать Windows-приложение "*Расчет удержаний с зарплаты*". Предусмотреть:

1. Создание входных форм:

Справочник работников (Табельный номер, ФИО работника, Код отдела, Количество льгот);

Начисления (Табельный номер, начислено).

2. На основе входных данных построить выходной документ:

Удержания (Табельный номер, Удержания в ПФ, Удержания ПН).

3. Выполнить расчет общей суммы удержаний по заданному табельному номеру.

Вариант 9

Создать Windows-приложение "*Формирование плана выпуска продукции*". Предусмотреть:

1. Создание входных форм:

Справочник продукции (Код, Наименование продукции, Цена сборки);

Справочник деталей (Код детали, Наименование, Цена);

Состав продукции (Код продукции, Код детали, Количество деталей в продукции).

2. На основе входных данных и планового количества продукции построить выходной документ:

План выпуска продукции (Код, Стоимость выпуска продукции).

3. Выполнить расчет стоимости выпуска заданной продукции.

Вариант 10

Создать Windows-приложение "*Учет выполнения работ в автосервисе* ". Предусмотреть:

1. Создание входных форм:

Справочник видов работ (Код работы, Марка автомобиля, Наименование

работы, Код исполнителя);

Справочник исполнителей работ (Код исполнителя, ФИО);

Заказы (Дата, Номер заказа, ФИО клиента, Марка автомобиля, Наименование работы, Код исполнителя).

2. На основе входных данных и заданных Номеров заказа построить выходной документ:

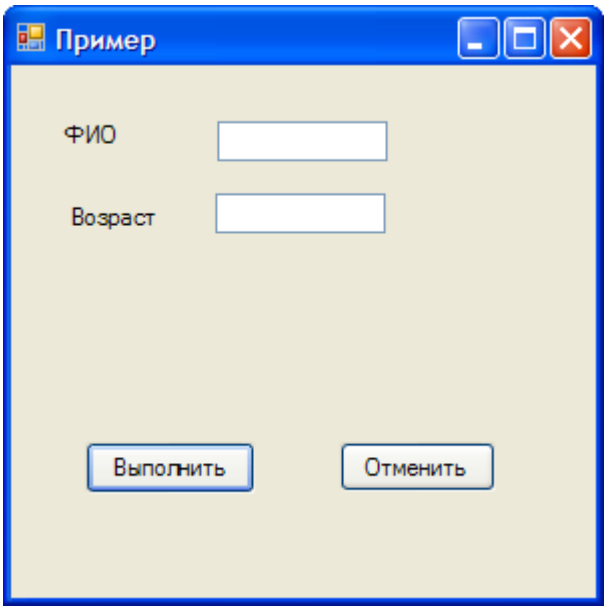
Выполнение работ (Код исполнителя, Номер заказа, Марка автомобиля,

Стоимость заказа).

3. Выполнить расчет стоимости заказа по заданному Номеру заказа.

Пример программного кода по практическому заданию 3:

1. Создать форму.
2. Занести данные в структуру.
3. Создать файл, содержащий записи: ФИО и возраст.
4. Вывести содержимое записей на экран.



Фрагмент текста работающей программы

```
usingSystem;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication4
{
struct stud
         {
        public
             stud (string fname, int fage)
        {
            name  = fname;
            age = fage;
        }
public      string name;
public      int age;
         }
    public partial class Form1 : Form
    {
public Form1()
        {
            InitializeComponent();
        }

    //============================
private void button1_Click(object sender,
EventArgs e)
        {
int k;
string str;
            stud z= new stud();
if (textBox1.Text == "")
            {
MessageBox.Show(" ВведитеФИОстудента");
return;
```

```
            }
        //string str = " Hello!!!";
            z.name = textBox1.Text;
if (textBox2.Text == "")
            {
     MessageBox.Show(" Введитевозрасткак
целоечисло");
return;
            }
            str = textBox2.Text;
            k = Int32.Parse(str);
            z.age= k;
            MessageBox.Show(" ФИО: "+z.name);
            str = z.name + " ";
MessageBox.Show(str+"возраст: "+k);
          }

private void button2_Click
(object sender, EventArgs e)
      {
        textBox2.Text = "";
        textBox1.Text = "";
        MessageBox.Show(" Выполненаотмена");
return;
      }
    }
```

Библиографический список

1. Фленов, Ф.Е. Библия C#. / Ф.Е. Фленов. – СПб.: БХВ-Петербург, 2009. – 560 с.
2. Шилдт, Герберт. C# 3.0: руководство для начинающих.: пер. с англ. / Герберт Шилдт. – М.: ООО «И.Д. Вильямс», 2009. – 668 с.
3. Программирование: Учебное пособие / Сост.:Е.О.Шумова. Национальный минерально-сырьевой университет «Горный». - СПб, 2013, 87 с. (Электронный вариант)
4. Объектно-ориентированное программирование: Учебное пособие / Сост.:.Е.О.Шумова. Национальный минерально-сырьевой университет «Горный». - СПб, 2013, 124 с. (Электронный вариант)
5. Рачева, Н.В., Сибирев В.Н.: Алгоритмирование ипрограммирование:учебно-методический комплекс / сост. Н.В.Рачева, В.Н.Сибирев. – СПб.: Изд-во СЗТУ, 2009. – 81 с.

СОДЕРЖАНИЕ