НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНА ТЕХНІКА ТА ПРОГРАМУВАННЯ»

*Звiт з лабораторної роботи №4*

*Тема: «Властивості класу. Обробка рядків. String Builder»*

Виконав:   
ст. гр. КІТ-119б  
Шарма О.Р.

Перевірив:  
Бартош М.В.

Харків – 2021

***Індивідуальне завдання:***

1. Для автоматичного розрахунку значень при читанні використовувати властивості "тільки для

читання "(без реалізації методу set).

2. Висновок для обраного студента назви групи (абревіатура назви факультету,

номер спеціальності, рік надходження, індекс).

3. Висновок для обраного студента номера курсу і семестру на поточний момент.

4. Розрахунок і висновок для обраного студента віку на поточний момент (до дня).

5. Продемонструвати ефективне використання StringBuilder для обробки рядків.

***Опис програми***

Дана програма містить 6 класів з методами.

**Клас MainMenu**

Головний клас програми, містить:

1. Метод Start

Функції що виконуються :

* 1. Створення контейнеру, який міститиме масив об’єктів типу Student.
  2. Запит у користувача чи бажає він додати студента.
  3. Якщо так, то запит користувача заповнити всі дані.
  4. Створення на основі даних об’єкта типу Student.
  5. Додання об’єкту до контейнеру.
  6. Виведення на екран вміст контейнера за допомогою foreach.
  7. Знаходження студента по індексу.
  8. Збереження введених даних у файлі для подальшого редагування.
  9. Відновлення раніше збережених даних з файлу.
  10. Пошук і редагування даних студента.
  11. Видалення даних особової справи студента.
  12. Виведення для обраного студента назви групи.
  13. Виведення для обраного студента номера курсу і семестру на поточний момент.
  14. Розрахунок і виведення для обраного студента віку на поточний момент (до дня).

**Клас Student**

Містить поля – інформація про студенів. Містить сеттери та геттери, а також метод toString та два конструктори.

**Клас StudentArray**

Наслідується від IEnumerable та IEnumerator. Містить поля – масив студентів, розмір масиву. Також містить методи GetEnumerator, MoveNext,

**Клас Input**

Відповідає за вхідні дані

**Клас Validator**

Відповідає за валідацію вхідних даних

ТЕКСТ ПРОГРАМИ

**Файл Program**

using System;

using System.IO;

using System.Text;

namespace Sharma04

{

class Program

{

static void Main(string[] args)

{

StringBuilder sb = new StringBuilder();

var studArr = new StudentArray();

bool flag = true;

string name, lastname, patronym, faculty, specialty;

string path = @"C:\Users\{Dean}\source\repos\Sharma04\Sharma04\saved.txt";

DateTime birthday, date;

int progress, number;

string index;

string text;

int choice;

while (flag)

{

Console.WriteLine("Что Вы хотите сделать?\n 1 - Добавить данные про студента\n 2 - Вывести на экран данные\n 3 - Записать данные в файл" +

"\n 4 - Прочитать данные из файла\n 5 - Найти элемент по индексу\n 6 - Удалить данные о студенте\n 7 - Редактировать данные студента" +

"\n 8 - Вывести название группы студента\n 9 - Вывести текущий курс и семестр студента\n 10 - Вывести текущий возраст студента\n" +

"11 - Выход");

choice = int.Parse(Console.ReadLine());

switch (choice)

{

case 1:

name = Input.EnterName("Имя");

lastname = Input.EnterName("Фамилию");

patronym = Input.EnterName("Отчество");

birthday = Input.EnterDate("День рождения");

date = Input.EnterDate("Дату поступления");

index = Input.EnterUniInfo("Индекс группы");

faculty = Input.EnterUniInfo("Факультет");

specialty = Input.EnterUniInfo("Специальность");

progress = Input.EnterPercents("Успеваемость");

var stud = new Student(name, lastname, patronym, birthday, date, index, faculty, specialty, progress);

studArr.Add(stud);

break;

case 2:

foreach (var student in studArr)

{

Console.WriteLine(student + " ");

}

break;

case 3:

try

{

using (StreamWriter sw = new StreamWriter(path, false, System.Text.Encoding.Default))

{

sw.WriteLine("Студенты: ");

}

foreach (var student in studArr)

{

using (StreamWriter sw = new StreamWriter(path, true, System.Text.Encoding.Default))

{

sw.WriteLine(student);

}

}

Console.WriteLine("Запись выполнена");

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

break;

case 4:

try

{

using (StreamReader sr = new StreamReader(path))

{

text = sr.ReadToEnd();

string[] separatingStrings = { " ", "\r", "\n", ":", "00:00:00", "\t", "Студенты", "Имя", "Фамилия", "Отчество", "День рождения",

"Дата поступления", "Индекс группы", "Факультет", "Специальность", "Успеваемость"};

string[] words = text.Split(separatingStrings, System.StringSplitOptions.RemoveEmptyEntries);

for (int i = 0; i < words.Length / 9; i++)

{

name = words[9 \* i];

lastname = words[9 \* i + 1];

patronym = words[9 \* i + 2];

birthday = DateTime.Parse(words[9 \* i + 3]);

date = DateTime.Parse(words[9 \* i + 4]);

index = words[9 \* i + 5];

faculty = words[9 \* i + 6];

specialty = words[9 \* i + 7];

progress = int.Parse(words[9 \* i + 8]);

stud = new Student(name, lastname, patronym, birthday, date, index, faculty, specialty, progress);

studArr.Add(stud);

}

}

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

break;

case 5:

Console.WriteLine("Номер студента, которого хотите найти: ");

number = int.Parse(Console.ReadLine());

studArr.Search(studArr, number);

break;

case 6:

Console.WriteLine("Номер студента, данные о котором хотите удалить: ");

number = int.Parse(Console.ReadLine());

studArr.Remove(studArr, number);

break;

case 7:

Console.WriteLine("Номер студента, данные о котором хотите отредактировать: ");

number = int.Parse(Console.ReadLine());

Console.WriteLine("Что хотите отредактировать? (1-имя, 2 - фамилию, 3 - отчество, 4 - день рождения, 5 - дата поступления, " +

"6 - индекс группы, 7 - факультет, 8 - специальность, 9 - успеваемость");

int n;

n = int.Parse(Console.ReadLine());

switch (n)

{

case 1:

studArr.Edit(studArr, number, "name");

break;

case 2:

studArr.Edit(studArr, number, "lastname");

break;

case 3:

studArr.Edit(studArr, number, "patronym");

break;

case 4:

studArr.Edit(studArr, number, "birthday");

break;

case 5:

studArr.Edit(studArr, number, "date");

break;

case 6:

studArr.Edit(studArr, number, "index");

break;

case 7:

studArr.Edit(studArr, number, "faculty");

break;

case 8:

studArr.Edit(studArr, number, "specialty");

break;

case 9:

studArr.Edit(studArr, number, "progress");

break;

}

break;

case 8:

Console.WriteLine("Номер студента, чью группу хотите узнать: ");

number = int.Parse(Console.ReadLine());

sb.Append(studArr.students[number - 1].Faculty);

sb.Append(studArr.students[number - 1].Specialty);

sb.Append("-");

sb.Append(studArr.students[number - 1].Date.Year);

sb.Append(studArr.students[number - 1].Index);

sb.AppendLine();

Console.WriteLine(sb.ToString());

break;

case 9:

Console.WriteLine("Номер студента, чей номер курса и семестра на текущий момент хотите узнать: ");

number = int.Parse(Console.ReadLine());

int course, semester;

course = DateTime.Today.Year - studArr.students[number - 1].Date.Year;

if (DateTime.Today.Month >= 7 && DateTime.Today.Month <= 12)

{

semester = course \* 2 - 1;

}

else

{

semester = course \* 2;

}

Console.WriteLine($"Курс : {course}, семестр : {semester}");

break;

case 10:

Console.WriteLine("Номер студента, чей текущий возраст хотите узнать: ");

number = int.Parse(Console.ReadLine());

DateTime today = DateTime.Today;

DateTime b = studArr.students[number - 1].Birthday;

TimeSpan old = today.Subtract(b);

var d = new DateTime(old.Ticks);

/\*int year = Convert.ToInt32(Math.Floor(old.TotalDays / 365));

int month = Convert.ToInt32(Math.Floor((old.TotalDays % 365) / 31));

int day = Convert.ToInt32(Math.Floor((old.TotalDays % 365) % 31));\*/

Console.WriteLine($"Возраст: {d.Year - 1} лет, {d.Month - 1} месяцев, {d.Day - 1} дней");

break;

case 11:

flag = false;

break;

}

}

}

}

}

**Файл Input**

using Microsoft.VisualBasic.CompilerServices;

using System;

using System.Collections.Generic;

using System.Text;

namespace DotNetLab1

{

class Input

{

private static string \_errorMessage = "Invalid input. Check and try again";

public static string EnterName(string fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

string name = Console.ReadLine();

while (!Validator.ValidateName(name))

{

Console.WriteLine(\_errorMessage);

name = Console.ReadLine();

}

return name;

}

public static string EnterUniInfo(string fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

string sentence = Console.ReadLine();

while (!Validator.ValidateSentence(sentence))

{

Console.WriteLine(\_errorMessage);

sentence = Console.ReadLine();

}

return sentence;

}

public static string EnterString(string fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

string str = Console.ReadLine();

return str;

}

public static int EnterPercents(string fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

while (true)

{

try

{

int value = Convert.ToInt32(Console.ReadLine());

if (Validator.ValidateIntByRange(0, 100, value))

{

return value;

}

continue;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

public static int EnterInt(string fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

while (true)

{

try

{

int value = Convert.ToInt32(Console.ReadLine());

return value;

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

}

}

public static DateTime EnterDate(String fieldName)

{

Console.WriteLine("Enter " + fieldName + ":");

while (true)

{

try

{

Console.WriteLine("Enter year:");

int year = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter month:");

int month = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter day:");

int day = Convert.ToInt32(Console.ReadLine());

return new DateTime(year, month, day);

}

catch (Exception e)

{

Console.WriteLine("An exception occured while date input:" + e.Message);

}

}

}

}

}

**Файл Student**

using Microsoft.VisualBasic.FileIO;

using System;

namespace Sharma04

{

public class Student

{

public string Name { get; set; }

public string Lastname { get; set; }

public string Patronym { get; set; }

public DateTime Birthday { get; set; }

public DateTime Date { get; set; }

public string Index { get; set; }

public string Faculty { get; set; }

public string Specialty { get; set; }

public int Progress { get; set; }

public Student() { }

public Student(string name, string lastname, string patronym, DateTime birthday, DateTime date, string index, string faculty, string specialty, int progress)

{

this.Name = name;

this.Lastname = lastname;

this.Patronym = patronym;

this.Birthday = birthday;

this.Date = date;

this.Index = index;

this.Faculty = faculty;

this.Specialty = specialty;

this.Progress = progress;

}

public override string ToString()

{

return $" Имя: {Name}\n Фамилия: {Lastname}\n Отчество: {Patronym}\n День рождения: {Birthday}\n Дата поступления: {Date}\n Индекс группы: {Index}\n Факультет: {Faculty}\n Специальность: {Specialty}\n Успеваемость: {Progress}\n \n";

}

}

}

**Файл StudentArray**

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Text;

namespace Sharma04

{

public class StudentArray : IEnumerable, IEnumerator

{

static int size = 1;

public Student[] students = new Student[size];

//Student[] students2 = new Student[size];

int index = -1;

int currentSize = 0;

// Реализуем интерфейс IEnumerable

public IEnumerator GetEnumerator()

{

return this;

}

// Реализуем интерфейс IEnumerator

public bool MoveNext()

{

if (index == students.Length - 1)

{

Reset();

return false;

}

index++;

return true;

}

public void Reset()

{

index = -1;

}

public object Current

{

get

{

return students[index];

}

}

public void Add(Student student)

{

currentSize = NewSize();

if (currentSize < size)

{

students[currentSize] = student;

}

else

{

size = currentSize + 1;

Array.Resize(ref students, size);

students[size - 1] = student;

}

}

int NewSize()

{

currentSize = 0;

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

{

if (students[i] != null)

currentSize++;

}

return currentSize;

}

public void Search(StudentArray records, int number)

{

Console.WriteLine("Студент по индексу :");

Console.Write(records.students[number - 1] + " ");

}

public void Remove(StudentArray records, int number)

{

currentSize = NewSize();

students[number - 1] = null;

for (int i = number; i < currentSize; i++)

{

if (students[i] != null)

{

students[i - 1] = students[i];

}

}

Array.Resize(ref students, currentSize - 1);

Console.WriteLine("Измененный список студентов :");

foreach (var student in records)

{

Console.WriteLine(student + " ");

}

}

public void Edit(StudentArray records, int number, string s)

{

switch (s)

{

case "name":

records.students[number - 1].Name = Input.EnterName("Введите имя");

break;

case "lastname":

records.students[number - 1].Lastname = Input.EnterName("Введите фамилию");

break;

case "patronym":

records.students[number - 1].Patronym = Input.EnterName("Введите отчество");

break;

case "birthday":

records.students[number - 1].Birthday = Input.EnterDate("Введите новую дату рождения");

break;

case "date":

records.students[number - 1].Date = Input.EnterDate("Введите новую дату поступления");

break;

case "index":

records.students[number - 1].Index = Input.EnterUniInfo("Введите индекс группы");

break;

case "faculty":

records.students[number - 1].Faculty = Input.EnterUniInfo("Введите индекс группы");

break;

case "specialty":

records.students[number - 1].Specialty = Input.EnterUniInfo("Введите специальность");

break;

case "progress":

records.students[number - 1].Progress = Input.EnterPercents("Введите академ.успеваемость в процентах");

break;

}

Console.WriteLine("Измененный список студентов :");

foreach (var student in records)

{

Console.WriteLine(student + " ");

}

}

}

}

**Файл Validator**

using System;

using System.Collections.Generic;

using System.Text;

using System.Text.RegularExpressions;

namespace Sharma04

{

class Validator

{

private static string \_namePattern = @"^[A-ЯЁ][а-яё]+$";

private static string \_sentencePattern = @"\b[^\d\W]+\b";

public static bool ValidateName(string name)

{

return Regex.IsMatch(name.Trim(), \_namePattern, RegexOptions.IgnoreCase);

}

public static bool ValidateSentence(string sentence)

{

return Regex.IsMatch(sentence.Trim(), \_sentencePattern, RegexOptions.IgnoreCase);

}

public static bool ValidateIntByRange(int min, int max, int value)

{

return value >= min && value <= max;

}

}

}

РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ

Результати роботи програми приведено нижче на скріншотах:

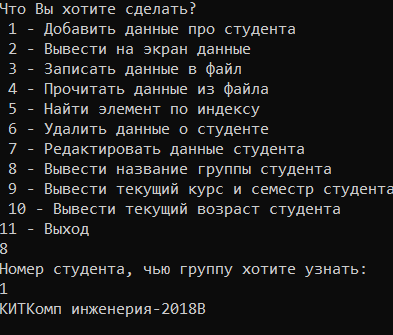


Рисунок 1 –функція «дізнатися індекс»

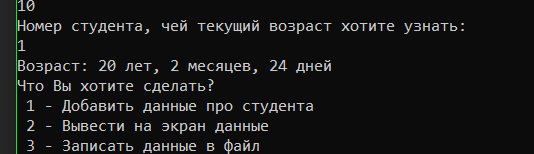


Рисунок 2 – вік введеного студента

Висновки:

В ході данної лабораторної роботи було доповнено програму з попередньої роботи, було додано методи для знаходження індексу групи студента, його курса та поточного віку.