МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №2

По дисциплине «Современные платформы программирования»

Специальность ПО-8

Выполнил:

Буртик Я.В.

студент группы ПО-8

Проверил:

ст. преп. кафедры ИИТ,

«\_\_»\_\_\_\_\_\_\_\_\_\_\_2024 г.

**Цель работы:** приобрести базовые навыки работы с файловой системой в С#

**Вариант 4**

**Задание 1. Напишите программу, которая выводит слова, располагая их в порядке убывания частоты их появления. Перед каждым словом должно быть число его появлений.**

**Код программы**

string path = "input.txt";  
Dictionary<string, int> dictionary = new Dictionary<string, int>();  
using (StreamReader reader = new StreamReader(path))  
{  
 string[] strings = reader.ReadToEnd().Split(',',' ','.');  
 foreach (var str in strings)  
 {  
 if (dictionary.ContainsKey(str))  
 {  
 dictionary[str]++;  
 }  
 else  
 {  
 dictionary.Add(str,1);  
 }  
 }  
}  
  
var ordered =   
 dictionary.OrderByDescending(x => x.Value).  
 ToDictionary(x => x.Key, x => x.Value);;  
  
  
foreach (var dictItem in ordered)   
{  
 Console.WriteLine($"{dictItem.Key} {dictItem.Value}");  
}

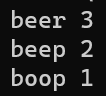
**Спецификация ввода**

input.txt

beep boop beer beer beer beep  
**Спецификация вывода**

**слово кол-во повторений**

**Пример**



**Задание 2. Утилита nl выводит переданный файл в стандартный вывод или в другой файл, выполняя нумерацию его строк. Если файл не задан или задан как –, читает стандартный ввод.**

**Формат использования: nl [-i] [-l] [-n] входной\_файл [выходной\_файл]**

**• -i ЧИСЛО Задает шаг увеличения номеров строк**

**• -l 1/0 Задает флаг нумерации пустых строк**

**• -n ФОРМАТ Использовать заданный формат для номеров строк.**

**ln – номер выравнен по левому краю, без начальных нулей**

**rn – номер выровнен по правому краю, без начальных нулей**

**rz – номер выровнен по правому краю с начальными нулямиКод программы**

int i = 1;  
int l = 1;  
string n = "ln";  
string inputPath;  
string outputPath = "-";  
if (args.Contains("-i"))  
{  
 int iIndex = Array.IndexOf(args, "-i");  
 i = Convert.ToInt32(args[iIndex + 1]);  
}  
  
if (args.Contains("-l"))  
{  
 int lIndex = Array.IndexOf(args, "-l");  
 l = Convert.ToInt32(args[lIndex + 1]);  
}  
  
if (args.Contains("-n"))  
{  
 int nIndex = Array.IndexOf(args, "-n");  
 n = args[nIndex + 1];  
}  
  
if (args.Length % 2 == 0)  
{  
 inputPath = args[^2];  
 outputPath = args[^1];  
}  
else  
{  
 inputPath = args[^1];  
}  
*/\*  
Console.WriteLine($"-i {i}\n" +  
 $"-l {l}\n" +  
 $"-n {n}\n" +  
 $"path1 {inputPath}\n" +  
 $"path2 {outputPath}");  
\*/*List<string> list = new List<string>();  
using (StreamReader reader = new StreamReader(inputPath))  
{  
 int index = 1;  
 string[] str = reader.ReadToEnd().Split("\n");  
 int count = str.Length;  
 for (int j = 0; j < str.Length; j++)  
 {  
 str[j] = str[j].Trim();  
 if (str[j] == "")  
 {  
 if (l == 1)  
 {  
 str[j] = AddIndex(str[j], index, count, n);  
 }  
 else  
 {  
 index -= i;  
 }  
 }  
 else  
 {  
 str[j] = AddIndex(str[j], index, count, n);  
 }  
 list.Add(str[j]);  
 index += i;  
 }  
}  
   
if (outputPath == "-")  
{  
 foreach (var str in list)  
 {  
 Console.WriteLine(str);  
 }  
}  
else  
{  
 using (StreamWriter writer = new StreamWriter(outputPath))  
 {  
 foreach (var str in list)  
 {  
 writer.WriteLine(str);  
 }  
 }  
}  
  
Console.ReadLine();  
  
string AddIndex(string str, int index, int count, string type)  
{  
 switch (type)  
 {  
 case "ln":  
 return $"{index}. {str}";  
 case "rn":  
 string s1 = "";  
 for (int j = 0; j < count.ToString().Length - index.ToString().Length ; j++)  
 {  
 s1 += " ";  
 }  
 return $"{s1}{index}. {str}";  
 case "rz":  
 string s2 = "";  
 for (int j = 0; j < count.ToString().Length - index.ToString().Length ; j++)  
 {  
 s2 += "0";  
 }  
 return $"{s2}{index}. {str}";  
 }  
  
 return "";  
}

**Спецификация ввода**

* nl аргументы

**Пример**



**Спецификация вывода**

Содержимое файла

**Пример**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Вывод**

Приобрели базовые навыки работы с файловой системой в С#.