Министерство образования Республики Беларусь  
Учреждение образования  
«Брестский государственный технический университет»  
Кафедра ИИТ

Лабораторная работа №2  
По дисциплине: «Современные системы программирования»

Выполнил:  
студент 3 курса  
группы ИИ-21(I)  
Кирилович А. А.

Проверила:  
Якимук А.В.

Брест 2023

**Цель работы:** освоить создание консольных приложений; освоить работу с файлами; научиться строить план тестирования приложения.

**Ход работы:**

**Задание 1:** Во входном текстовом файле хранится числовой массив объёмом до 1 млн значений. Сами значения являются целыми положительными и не превосходят 1000. Возможны повторяющиеся значения. В выходной текстовый файл нужно вывести эти числа в порядке возрастания.

Структура входного файла:

первая строка в файле показывает длину числового массива (от 1 до 1000000).

Далее идет указанное количество числовых значений от 1 до 1000 каждое по одному числу в строке.

Решение должно быть выполнено в виде консольного приложения, результаты выведены в текстовый файл.

**Задание 2:** Модифицируйте программу, чтобы она выводила данные в порядке убывания.

**Задание 3:** Модифицируйте программу, чтобы она могла по требованию пользователя выводить данные либо по возрастанию, либо по убыванию.

**Код программы со всеми модификациями:**

/\*use std::fs::File;

use std::io::{self, Write};

use rand::Rng;

fn main() -> io::Result<()> {

let count = rand::thread\_rng().gen\_range(1..=1000000);

let mut file = File::create("random\_numbers.txt")?;

write!(file, "{}\n", count)?;

for \_ in 0..count {

let num = rand::thread\_rng().gen\_range(1..=1000);

write!(file, "{}\n", num)?;

}

println!("The file 'random\_numbers.txt' was successfully created and filled in.");

Ok(())

}\*/

use std::env;

use std::error::Error;

use std::fs::File;

use std::io::{BufRead, BufReader, Write};

fn main() -> Result<(), Box<dyn Error>> {

let args: Vec<String> = env::args().collect();

if args.len() != 3 {

eprintln!("Usage: {} <input\_file> <sort\_order>", args[0]);

std::process::exit(1);

}

let input\_file = &args[1];

let sort\_order = &args[2];

let file = File::open(input\_file)?;

let reader = BufReader::new(file);

let mut lines = reader.lines();

let \_ = lines.next();

let mut numbers: Vec<u32> = lines

.filter\_map(|line| line.ok())

.filter\_map(|line| line.parse().ok())

.collect();

match sort\_order.as\_str() {

"asc" => numbers.sort(),

"desc" => numbers.sort\_by(|a, b| b.cmp(a)),

\_ => {

eprintln!("Invalid collation: {}", sort\_order);

std::process::exit(1);

}

}

let output\_file\_name = "output.txt";

let mut output\_file = File::create(output\_file\_name)?;

for num in &numbers {

writeln!(output\_file, "{}", num)?;

}

println!("The data was successfully sorted and written to the file '{}'",

output\_file\_name);

Ok(())

}

**Вывод:** в ходе данной лабораторной работы я научился создавать консольные приложения для работы с файлами.