МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Программирование»
Тема: Условия, циклы, оператор switch

Студент гр. 9381 _____ Колованов Р.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2019

Цель работы.

Изучение основ языка С и получение навыков работы с основными управляющими конструкциями, типами данных, функциями стандартной библиотеки ввода/вывода.

Задание.

Вариант 6.

Напишите программу, выделив каждую подзадачу в отдельную функцию. Реализуйте программу, на вход которой подается одно из значений 0, 1, 2, 3 и массив целых чисел размера не больше 100. Числа разделены пробелами. Строка заканчивается символом перевода строки. В зависимости от значения, функция должна выводить следующее:

0: Индекс первого отрицательного элемента. (index_first_negative)

1: Индекс последнего отрицательного элемента. (index_last_negative)

2: Найти сумму модулей элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент). (sum_between_negative)

3: Найти сумму модулей элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент). (sum_before_and_after_negative)

иначе необходимо вывести строку "Данные некорректны".

Выполнение работы.

Сначала пользователь вводит число (0-3), в зависимости от значения которого программа будет выводить: 0 – индекс первого отрицательного элемента, 1 – индекс последнего отрицательного элемента, 2 – сумма модулей элементов между этими индексами (включая первый и не включая последний отрицательные элементы), 3 – сумма модулей элементов, которые не попадают в промежуток из подзадачи 2. При помощи функции *scanf()* считываем

введенное пользователем число и записываем его в переменную ***action*** (нужна для хранения выбранной подзадачи). Далее программа попадает в цикл *do-while*, в котором будет обрабатываться последующие введенные пользователем числа. В данном цикле мы заполняем массив ***array*** (нужен для хранения входных данных; размер массива равен 100, так как по условию пользователь не введет больше 100 чисел) входными данными: в цикле начинаем считывать число и следующий символ после числа. Число записываем в массив, символ – в переменную ***nextSymbol*** (хранит следующий символ после введенного числа), а переменную ***arraySize*** (хранит количество заполненных элементов массива или просто количество входных данных) увеличиваем на единицу. Производим запись значений в массив до тех пор, пока следующий символ после числа не равен символу перевода на новую строку. После окончания ввода воспользуемся оператором switch для выбора нужной подзадачи (в зависимости от значения переменной ***action*** выбираем нужную нам функцию). Записываем результат выполнения выбранной функции в переменную ***answer*** (хранит ответ на задачу). Если answer будет равен -1, значит переданные в функцию входные данные некорректны, следовательно выводим “Данные некорректны”. Иначе выводим переменную ***answer*** на экран.

Программа содержит 4 функции, каждая из которых предназначена для выполнения определенной подзадачи:

1) *int getFirstNegativeNumberIndex(int* array, int arraySize);*

   На вход принимает указатель на массив данных и его размер. Используется для нахождения индекса первого отрицательного элемента в массиве входных данных. Возвращает индекс найденного элемента, а если элемент не найден – возвращает -1. Принцип работы: пробегаемся по элементам с начала массива до тех пор, пока не найдем отрицательный элемент.

2) *int getLastNegativeNumberIndex(int* array, int arraySize);*

   На вход принимает указатель на массив данных и его размер. Используется для нахождения индекса последнего отрицательного

элемента в массиве входных данных. Возвращает индекс найденного элемента, а если элемент не найден – возвращает -1. Принцип работы: пробегаемся по элементам с конца массива до тех пор, пока не найдем отрицательный элемент.

3) *int getSumBetweenNegativeNumbers(int* array, int arraySize);*

   На вход принимает указатель на массив данных и его размер. Используется для нахождения суммы модулей элементов между первым (включая) и последним (не включая) отрицательными элементами. Возвращает найденную сумму, а если в массиве нет первого или последнего отрицательного элемента – возвращает -1. Принцип работы: пробегаемся между первым (включая) и последним (не включая) отрицательными элементами, добавляя каждый раз в переменную ***sum*** модуль значения текущего элемента.

4) *int getSumBeforeAndAfterNegativeNumbers(int* array, int arraySize);*

   На вход принимает указатель на массив данных и его размер. Используется для нахождения суммы модулей элементов между началом массива и первым (не включая) отрицательным элементом, последним (включая) отрицательным элементом и концом массива. Возвращает найденную сумму, а если в массиве нет первого или последнего отрицательного элемента – возвращает -1. Принцип работы: пробегаемся между началом массива и первым (не включая) отрицательным элементом, последним (включая) отрицательным элементом и концом массива, добавляя каждый раз в переменную ***sum*** модуль значения текущего элемента.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	3	
2.	1 8 0 -3 7 -3 9 223 0 0 -1 -1 -1	11	
3.	3 0 -9 34 -1 2 -1 3 4 5 -9 -1	1	

Выводы.

Были изучены основы языка С; получены навыки работы с основными управляющими конструкциями, типами данных, функциями стандартной библиотеки ввода/вывода.

Разработана программа, на вход которой подается одно из значений 0-3 и массив целых чисел размера не больше 100. В зависимости от введенного пользователем значения, программа может выводить следующее:

- Индекс первого отрицательного элемента.
- Индекс последнего отрицательного элемента.
- Сумму модулей элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент).
- Сумму модулей элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент).

Для обработки команд пользователя использовались условные операторы *if-else,* оператор switch. Также в программе использовались функции для разделения функционала программы на подпрограммы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <stdlib.h>

int getFirstNegativeNumberIndex(int* array, int arraySize) {
    for (int i = 0; i < arraySize; i++)
        if (array[i] < 0)
            return i;

    return -1;
}

int getLastNegativeNumberIndex(int* array, int arraySize) {
    for (int i = arraySize - 1; i >= 0; i--)
        if (array[i] < 0)
            return i;

    return -1;
}

int getSumBetweenNegativeNumbers(int* array, int arraySize) {
    int sum = 0;
    int begin = getFirstNegativeNumberIndex(array, arraySize);
    int end = getLastNegativeNumberIndex(array, arraySize);

    if (begin == -1 || end == -1)
        return -1;

    for (int i = begin; i < end; i++)
        sum += abs(array[i]);

    return sum;
}

int getSumBeforeAndAfterNegativeNumbers(int* array, int arraySize) {
    int sum = 0;
    int begin = getFirstNegativeNumberIndex(array, arraySize);
    int end = getLastNegativeNumberIndex(array, arraySize);

    if (begin == -1 || end == -1)
        return -1;

    for (int i = 0; i < begin; i++)
        sum += abs(array[i]);

    for (int i = end; i < arraySize; i++)
        sum += abs(array[i]);

    return sum;
}

int main() {
    int array[100] = { 0 }, arraySize = 0, action = 0, answer = 0;
```

```
    char nextSymbol = ' ';

    scanf("%d", &action);

    do {
        int value;

        scanf("%d", &value);
        scanf("%c", &nextSymbol);

        array[arraySize] = value;
        arraySize++;

    } while (nextSymbol != '\n');

    switch (action) {
        case 0:
            answer = getFirstNegativeNumberIndex(array, arraySize);
            break;
        case 1:
            answer = getLastNegativeNumberIndex(array, arraySize);
            break;
        case 2:
            answer = getSumBetweenNegativeNumbers(array, arraySize);
            break;
        case 3:
            answer = getSumBeforeAndAfterNegativeNumbers(array, arrayS
ize);
            break;
        default:
            answer = -1;
    }

    if (answer == -1) {
        printf("Данные некорректны");
    } else {
        printf("%d", answer);
    }

    return 0;
}
```