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. 1304 _____ Дешура Д.В.

Преподаватель _____ Чайка К.В.

Санкт-Петербург
2021

Цель работы.

Создать программу, позволяющую ввести массив из чисел и обработать его одним из 4-х заданных способов. Закрепить навыки работы с условиями, циклами и оператором switch.

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.

n- целое число, = 100, определяет длинну массива чисел Data_arr[];

number_of_command- целое число, хранит номер команды;

flag_2- целое число, = 0, считает количество введёных отрицательных чисел, если таких меньше 2-х, то введённые данные не корректны;

c- целое число,;

result_of_programm- целое число, результат программы;

Data_arr[]- массив целых чисел, хранит данные для обработки;

ch- символ, во время ввода считывает пробелы между числами, как только примнимает знак переноса строки, ввод прерывается;

i- целое число, счётчик цикла for;

int main() - основная функция программы. Считывает ввод: сначала первое число в переменную *number_of_command*, затем все оставшиеся числа по одному в массив *Data_arr[]*, пока не встретится ‘\n’, паралельно заполнению массива, считаем количество отрицательных среди чисел и записываем результат в переменную в переменную *flag_2*. Далее – проверка данных, если *flag_2* < 2 (колличество отрицательных чисел меньше 2), то выводим “Данные некорректны” и завершаем прграмму. Затем в зависимости от введённого *number_of_command*, вызываем функции и присваиваем их значения переменной *result_of_programm*. Действия в зависимости от значения переменной *number_of_command*: index_first_negative() для 0, index_last_negative() для 1, sum_between_negative() для 2, sum_before_and_after_negative() для 3, для любого другого значения выводим “Данные некорректны” и завершаем прграмму.
вызывает функции index_first_negative(), index_last_negative(), sum_between_negative(), sum_before_and_after_negative(); выводит результат программы.

int index_first_negative(int Data_arr[], int n) - функция, вызываемая при *number_of_command* == 0, принимает массив целых чисел *Data_arr* и его размер *n*. Запускает цикл *for(int i = 0; i < n; i++)* перебирает элементы массива, встретив первый отрицательный элемент, возвращает его индекс.

int index_last_negative(int Data_arr[], int n) - фунция, вызываемая при *number_of_command* == 1, принимает массив целых чисел *Data_arr* и его размер *n*. Вводит целочисленную переменную *result*. Запускает цикл *for(int i = 0; i < n; i++)* перебирает элементы массива, встретив отрицательный элемент, передаёт его индекс переменной *result*, таким образом после прохождения цикла *result* будет хранить индекс последнего такого числа. Возвращает переменную result.

int sum_between_negative(int Data_arr[], int index_first_negative, int index_last_negative) - функция, вызываемая при *number_of_command* == 2,

принимает массив целых чисел *Data_arr*, функцию *index_first_negative(Data_arr, n)* и функцию *index_last_negative(Data_arr, n)*. Вводит целочисленную переменную *sum* = 0, запускает цикл *for(int index_arr = index_first_negative; index_arr < index_last_negative; index_arr++)*. Цикл проходит от первого отрицательного (включительно), до последнего отрицательного (не включительно), увеличивая *sum* на модуль каждого элемента. После прохода цикла функция возвращает *sum*.

int sum_before_and_after_negative(int Data_arr[], int index_first_negative, int index_last_negative, int n) - функция, вызываемая при *number_of_command* == 3, принимает массив целых чисел *Data_arr*, его размер *n*, функцию *index_first_negative(Data_arr, n)* и функцию *index_last_negative(Data_arr, n)*. Вводим целочисленную переменную *sum*. Функция состоит из двух циклов. Первый - *for(int index_arr = 0; index_arr < index_first_negative; index_arr++)* проходит по массиву данных от начала до первого отрицательного элемента (не включительно), прибавляя к переменной *sum* модуль каждого элемента. Второй цикл *for(int index_arr = index_last_negative; index_arr < n; index_arr++)* выполняет те же действия, но двигается от последнего отрицательного элемента (включительно) к концу массива. Функция возвращает значение переменной *sum*.

Разработанный программный код см. в приложении А.

Тестирование.

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	Пройден
2.	1 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	20	Пройден
3.	2 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	226	Пройден
4.	3 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	30	Пройден
5.	4 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	Данные некорректны	Пройден
6.	2 1 16 2 18 -22 15 3 13 0 6 1 9 24 1 18 15 28 20 17 16 11	Данные некорректны	Пройден

Выводы.

Выполнив лабораторную работу №1, мы создали программу, позволяющую ввести массив из чисел и обработать его одним из 4-х заданных способов.

Были изучены основные управляющие конструкции языка С. Была разработана программа, выполняющая считывание с клавиатуры исходных данных. Для обработки команд пользователя использовался оператор *switch*.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: 1304_PR_Дешура_ДВ_ЛР1.c

```
#include <stdio.h>

int index_first_negative(int Data_arr[], int n){
    for(int i = 0; i < n; i++){
        if (Data_arr[i] < 0){
            return i;
        }
    }
}

int index_last_negative(int Data_arr[], int n){
    int result;
    for(int i = 0; i < n; i++){
        if (Data_arr[i] < 0){
            result = i;
        }
    }
    return result;
}

int sum_between_negative(int Data_arr[], int index_first_negative,
int index_last_negative){
        int sum = 0;
        for(int index_arr = index_first_negative; index_arr <
index_last_negative; index_arr++){
            if(Data_arr[index_arr] < 0){
                sum = sum - Data_arr[index_arr];
            }else{
                sum = sum + Data_arr[index_arr];
            }
        }

        return sum;
}

int sum_before_and_after_negative(int Data_arr[], int
index_first_negative, int index_last_negative, int n){
        int sum = 0;
        for(int index_arr = 0; index_arr < index_first_negative;
index_arr++){
            if(Data_arr[index_arr] < 0){
                sum = sum - Data_arr[index_arr];
            }else{
                sum = sum + Data_arr[index_arr];
            }
        }
        for(int index_arr = index_last_negative; index_arr < n;
index_arr++){
            if(Data_arr[index_arr] < 0){
                sum = sum - Data_arr[index_arr];
            }else{
                sum = sum + Data_arr[index_arr];
```

```
            }
        }
        return sum;
    }

    int main(){
        int number_of_command, result_of_programm, flag_2 = 0, c = 0, n
= 100;
        char ch;
        scanf("%d%c", &number_of_command, &ch);
        int Data_arr[n];

        for(int index_arr = 0; index_arr < n; index_arr++){
            scanf("%d%c", &Data_arr[index_arr], &ch);
            if(Data_arr[index_arr] < 0){
                flag_2++;
            }
            if(ch == '\n'){
                c = index_arr;
                break;
            }
        }

        for(int index_arr = c + 1; index_arr < n; index_arr++){
            Data_arr[index_arr] = 0;
        }

        if(flag_2 < 2){
            printf("Данные некорректны");
            return 0;
        }

        switch(number_of_command){
            case 0:
                result_of_programm = index_first_negative(Data_arr, n);
                break;
            case 1:
                result_of_programm = index_last_negative(Data_arr, n);
                break;
            case 2:
                result_of_programm   =   sum_between_negative(Data_arr,
index_first_negative(Data_arr, n), index_last_negative(Data_arr, n));
                break;
            case 3:
                result_of_programm                                    =
sum_before_and_after_negative(Data_arr, index_first_negative(Data_arr, n),
index_last_negative(Data_arr, n), n);
                break;
            default:
                printf("Данные некорректны");
                return 0;
        }

        printf("%d\n", result_of_programm);

        return 0;
    }
```