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. 3344		Тукалкин.В.А
Преподаватель		Глазунов.С.А

Санкт-Петербург
2023

Цель работы

Освоить работы с управляющими конструкциями на языке Си на примере использующей их программы.

Задание.

Вариант 4.

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 массиве есть хотя бы один четный и нечетный элемент.

В зависимости от значения, функция должна выводить следующее:

0: индекс первого чётного элемента. (index_first_even).

1: индекс последнего нечётного элемента. (index_last_odd).

2: Найти сумму модулей элементов массива, расположенных от первого чётного элемента и до последнего нечётного, включая первый и не включая последний. (sum_between_even_odd).

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

Иначе необходимо вывести строку "Данные некорректны".

Выполнение работы

Выполнение работы будет расписано по шагам:

1) Включить библиотеки stdio.h и stdlib.h

2) Написать функцию main

3) Объявить переменную first с типом int и считать первое значение (0,1,2 или 3)

4) Написать оператор switch, который будет выводить результат определённых функций в зависимости от значения first

5) Написать функцию index_first_even, в которую будет направляться массив и находить индекс первого чётного числа в массиве.

6) Тело функции index_first_even:

Объявить переменную first_even типа int и присвоить значение -1, для проверки. Далее условный цикл for, который считает индекс первого чётного числа в массиве. После индекс присваивается переменной first_even и используется break, чтобы first_even имел индекс первого чётного числа. В конце функции переменная first_even проверятся на изменение и возвращается с помощью return.

7) Написать функцию index_last_odd, в которую будет направляться массив и находить индекс последнего нечётного числа в массиве.

8) Тело функции index_last_odd:

Объявить переменную last_odd типа int и присвоить значение -1, для проверки. Далее условный цикл for, который считает индекс последнего нечётного числа в массиве. После индекс присваивается переменной last_odd. В конце функции переменная first_even проверятся на изменение и возвращается с помощью return.

9) Написать функцию sum_before_even_and_after_odd, в которую будет направляться массив и находить сумму элементов по модулю расположенных от первого чётного и до последнего нечётного, включая первый и не включая последний.

10) Тело функции sum_before_even_odd:

Объявить переменную sum с типом int и присвоить значение 0, объявить переменную first_even с типом int и присвоить значение index_first_even, объявить переменную last_odd с типом int и присвоить значение index_last_odd. После написать цикл while, условием которого будет first_even меньше last_odd, по выполнении условия переменной sum будет прибавляться модульное значение от элемента массива с индексом first_even и переменная first_even будет увеличиваться на 1. В конце возвращается переменная.

11) Написать функцию sum_before_even_and_after_odd, в которую будет направляться массив и находить сумму элементов по модулю расположенных от первого чётного и до последнего нечётного, включая первый и не включая последний.

12) Тело функции sum_before_even_and_after_odd:

Объявить переменную sum с типом int и присвоить значение 0, объявить переменную first_even с типом int и присвоить значение index_first_even, объявить переменную last_odd с типом int и присвоить значение index_last_odd. После написать условный оператор for и найти сумму всех модульных значений чисел до первого чётного числа, не включая его. Далее при помощи цикл while найти сумму всех модульных значений после последнего нечётного числа, включая его. Вернуть значение sum.

13) Создать массив array с типом int и длиной 100, после считать все значения в него оператором for, затем пустые ячейки заполнить нулями.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 2 3 4 5 6 7 8	1	Верный ответ
2.	1 2 3 4 5 6 7 8 9	7	Верный ответ
3.	2 37 10 -32 35 44 54 -9 0	175	Верный ответ
4.	3 37 10 -32 35 44 54 -9 0	46	Верный ответ
5.	4 2 24 -24 53 95 0 12 -4	Данные некорректны	Верный ответ

Выводы

Были изучены основные управляющие конструкции языка Си на примере использующей их программы.

Разработана программа, выполняющая операции с поступающим массивом чисел. На вход подаётся число и массив целых чисел, при помощи цикла for производится запись чисел и заполнение остального пространства массива “0”, для с помощью циклов for и операторов if находятся индексы первого чётного и последнего нечётного чисел, и операторов while находились суммы модулей между этими числами.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb1.c

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

int index_first_even(int array[100]){
        int first_even=-1;
        for(int i=0;i<100;i++){
                if(array[i]%2 == 0){
                        first_even=i;
                        break;
                }
        }
        if(first_even != -1){
                return first_even;
        }
}

int index_last_odd(int array[100]){
        int last_odd=-1;
        for(int i=0;i<100;i++){
                if(array[i]%2 != 0){
                        last_odd = i;
                }
        }
        if(last_odd != -1){
                return last_odd;
        }
}

int sum_between_even_odd(int array[100]){
        int   sum=0,   first_even   =   index_first_even(array),
last_odd=index_last_odd(array);
        while(first_even < last_odd){
                sum +=abs(array[first_even]);
                first_even++;
        }
        return sum;
}

int sum_before_even_and_after_odd(int array[100]){
        int sum=0, first_even = index_first_even(array), last_odd
= index_last_odd(array);
        for(int i=0;i<first_even;i++){
                sum+= abs(array[i]);
        }
        while(last_odd < 100){
                sum+= abs(array[last_odd]);
                last_odd++;
        }
        return sum;
}
```

```
int main(){
        int first=0;
        int i=0;
        scanf("%d",&first);
        char ch;
        int array[100];
        for(i = 0;(ch=getchar()) != '\n'; i++){
                scanf("%d",&array[i]);
        }
        for(;i<100;i++){
                array[i] = 0;
        }
        switch (first){
                case 0:
                        printf("%d\n",index_first_even(array));
                        break;
                case 1:
                        printf("%d\n",index_last_odd(array));
                        break;
                case 2:

printf("%d\n",sum_between_even_odd(array));
                        break;
                case 3:

printf("%d\n",sum_before_even_and_after_odd(array));
                        break;
                default:
                        printf("Данные некорректны\n");
                        break;
        }

}
```