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. 0382 _____ Диденко Д.В.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Изучение базовых управляющих конструкций языка Си.

Задание.

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 чётного элемента.

1 : индекс последнего нечётного элемента.

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

3 : Найти сумму модулей элементов массива, расположенных до первого чётного элемента (не включая элемент) и после последнего нечётного (включая элемент). Иначе необходимо вывести строку "Данные некорректны".

Основные теоретические положения.

Для выполнения данного задания потребовались встроенные функции языка С *printf()* и *scanf()* из библиотеки *stdio.h* для ввода и вывода данных соответственно и функция *abs()* из библиотеки *stdlib.h* для нахождения модуля числа, а также основные управляющие конструкции: *if()…else(), switch(), for(),while()*.

Выполнение работы.

Исходный код решения задачи см.в приложении А.

Подключаются библиотеки – *stdio.h* для ввода и вывода данных через функции *scanf()* и *printf()* соответственно и *stdlib.h* для нахождения модуля через функцию *abs()*. В главной функции программы *main()* объявляется целочисленный массив *arr* размерностью 100, целочисленная переменная

arr_size, задача которой – показывать программе текущую заполненность массива значимыми элементами, целочисленная переменная *V* (version), отвечающая за дальнейшие действия программы над массивом *arr*. С помощью функции *scanf()* записываем значение в переменную *V*,объявляется переменная *S* типа *char*, следящая за введением данных в массив (если значение «пробел», ввод продолжается, иначе - прекращается).В следующих двух строках через цикл *while* с условиями *arr_size < 100* и *S == ' '* заполняется массив *arr*. В условном операторе *switch* проверяется переменная *V*: в случае 0 – выводится на консоль индекс первого четного элемента массива с помощью функции *index_first_even*, в случае 1 – индекс последнего нечетного элемента массива с помощью функции *index_last_odd*, в случает 2 - сумма модулей элементов массива, расположенных от первого чётного элемента и до последнего нечётного, включая первый и не включая последний с помощью функции *sum_between_even_odd*, в случае 3 - сумма модулей элементов массива, расположенных до первого чётного элемента (не включая элемент) и после последнего нечётного (включая элемент) с помощью функции *sum_before_even_and_after_odd*, при любом другом значении *V* выводится «Данные некорректны».

Функция *index_first_even* с параметрами *int A[], int* n принимает в качестве аргументов массив *arr* и переменную *arr_size*. В теле функции объявляется целочисленная переменная *i = 0*, отвечающая за индекс элемента массива. Цикл *while* поочередно проверяет каждый элемент массива на четность с помощью условного оператора *if* , и заканчивает свою работу, если встречает четное число (*if(A[i]%2 == 0)*) с помощью оператора *break*, если же такое число не находится, то производится операция *i++* (добавление единицы) и цикл повторяется. Функция возвращает значение *i* –индекс первого четного элемента массива.

Функция *index_last_odd* с параметрами *int A[], int* n принимает в качестве аргументов массив *arr* и переменную *arr_size*. В теле функции объявляется целочисленная переменная *i = n-1*, отвечающая за индекс элемента массива (в

данном случае логичней начинать с последнего элемента массива, имеющего индекс n-1). Цикл *while* поочередно проверяет каждый элемент массива на четность с помощью условного оператора *if* , и заканчивает свою работу, если встречает нечетное число (*if(abs(A[i])%2 == 1)*) с помощью оператора *break*, если же такое число не находится, то производится операция *i--* (вычитание единицы) и цикл повторяется. Функция возвращает значение *i* –индекс последнего нечетного элемента массива.

Функция *sum_between_even_odd* с параметрами *int A[], int* n принимает в качестве аргументов массив *arr* и переменную *arr_size*.В теле функции объявляются целочисленные переменные *Begin* и *End*, получающие значения из функций *index_first_even* и *index_last_odd* соответственно, которые получают аргументы *A,n* равные массиву *arr* и переменной *arr_size*. Целочисленная переменная *summ = 0*, в которую с помощью цикла *for* будет записана сумма модулей элементов массива с *Begin* включительно до *End* невключительно. Функция возвращает значение *summ.*

Функция *sum_before_even_and_after_odd* с параметрами *int A[], int* n принимает в качестве аргументов массив *arr* и переменную *arr_size*. В теле функции объявляется целочисленная переменная *total_sum = 0*, в которую будет записана сумма модулей всех элементов массива с помощью цикла *for*. Функция возвращает результат операции *total_sum - sum_between_even_odd(A,n),* равный сумме модулей элементов массива, расположенных до первого чётного элемента (не включая элемент) и после последнего нечётного (включая элемент).

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 -8 -23 -30 -11 -28 15 -20 -24 -27 5 -13 5 21 -5 16 30 -	0	Программа работает верно

	12 15 -14 -28 -27 -11 -5 4 29 -5\n		
2.	1 -8 -23 -30 -11 -28 15 -20 -24 -27 5 -13 5 21 -5 16 30 -12 15 -14 -28 -27 -11 -5 4 29 -5\n	25	Программа работает верно
3.	2 -8 -23 -30 -11 -28 15 -20 -24 -27 5 -13 5 21 -5 16 30 -12 15 -14 -28 -27 -11 -5 4 29 -5\n	426	Программа работает верно
4	3 -8 -23 -30 -11 -28 15 -20 -24 -27 5 -13 5 21 -5 16 30 -12 15 -14 -28 -27 -11 -5 4 29 -5\n	5	Программа работает верно
5	4 -8 -23 -30 -11 -28 15 -20 -24 -27 5 -13 5 21 -5 16 30 -12 15 -14 -28 -27 -11 -5 4 29 -5\n	«Данные некорректны»	Программа работает верно
6	0 5 -7 -73 83 -8 5 8 7 4 2 -16 -17 8 2 6\n	4	Программа работает верно
7	1 5 -7 -73 83 -8 5 8 7 4 2 -16 -17 8 2 6\n	11	Программа работает верно
8	2 5 -7 -73 83 -8 5 8 7 4 2 -16 -17 8 2 6\n	50	Программа работает верно
9	3 5 -7 -73 83 -8 5 8 7 4 2 -16 -17 8 2 6\n	201	Программа работает верно

Выводы.

Были изучены базовые управляющие конструкции языка Си.

Разработана программа, обрабатывающая последовательность введенных чисел и выводящая на консоль результат (зависит от значения переменной V). Для ввода и вывода данных использовались функции *scanf()* и *printf()* библиотеки *stdio.h*. Функция abs() библиотеки *stdlib.h* возвращала модули чисел. Все данные обрабатывались с помощью функций, созданных непосредственно в программе. Использовались базовые управляющие конструкции языка Си: условные операторы *if(){}...else if(){}…else{}, switch(){case 1:… case 2… default:},* циклы *for(){}, while(){}.*

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb1.c

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

int index_first_even(int A[],int n){
      int i = 0;
      while (i<n){
            if(A[i]%2 == 0){
                  break;
            }else
                  i++;
      }
      return i;
}

int index_last_odd(int A[],int n){
      int i = n-1;
      while (i>=0){
            if (abs(A[i])%2 == 1){
                  break;
            }else
                  i--;
      }
      return i;
}

int sum_between_even_odd(int A[],int n){
      int Begin = index_first_even(A,n);
      int End = index_last_odd(A,n);
      int summ = 0;
      for (Begin; Begin < End; Begin++){
            summ += abs(A[Begin]);
      }
      return summ;
}

int sum_before_even_and_after_odd(int A[],int n){
      int total_sum = 0;
      for(int i = 0; i < n;i++){
            total_sum += abs(A[i]);
      }
      return total_sum - sum_between_even_odd(A,n);
}

int main(){
      int arr[100];
      int arr_size = 0;
      int V;
      scanf("%d", &V);
      char S = ' ';
      while (arr_size < 100 && S == ' '){
            scanf("%d%c",&arr[arr_size++],&S);}
      switch (V){
```

```
            case 0:
            printf("%d\n", index_first_even(arr,arr_size));
            break;
            case 1:
            printf("%d\n", index_last_odd(arr,arr_size));
            break;
            case 2:
            printf("%d\n", sum_between_even_odd(arr,arr_size));
            break;
            case 3:
            printf("%d\n", sum_before_even_and_after_odd(arr,arr_size));
            break;
            default:
            printf("Данные некорректны\n");
            break;
      }
      return 0;
}
```