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

Студентка гр. 0382 _____ Чегодаева Е.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Цель работы.

Изучение базовых управляющих конструкций языки Си.

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 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".

Основные теоретические положения.

Из библиотеки stdio.h: функции scanf() и printf() (Ввод и вывод соответственно).

Из библиотеки stdlib.h: функция abs() (Модуль числа).

Операторы: if(){} else{}, for (){}, while(){} , switch(){}.

Выполнение работы.

В функции *main{}* с помощью функции *skanf{}* присваивается целочисленное значение переменной *n*, объявленной ранее. Значение переменной *n* определяет то, к какой функции будет обращение в дальнейшем (или вывод "Данные некорректны"). Затем в теле функции объявляется целочисленный массив *arr* размером *N (N = 100),* переменная *len*, которая хранит количество элементов массива (изначально *len* = 0), символьная переменная *gap= “ “*. Далее вводится сам массив посредствам цикла *while(){},* где с помощью функции *skanf{}* вводится каждый целочисленный элемент массива *arr[i]* и идёт

счёт элементов этого массива. После этого оператор *switch(n){}* выполняет различные команды, зависящие от значения переменной *n*.

При *n* = 0: происходит обращение к функции *index_first_even(arr, len){}*.Функция получает на вход массив *arr* и значение переменной *len.* Происходит перебор каждого элемента массива посредствам цикла *for(){}* и поиск первого чётного элемента массива (остаток от деления на 2 равен 0). Для того, что бы отрицательные элементы массива так же рассматривались в переборе используется функция *abs().* Функция *index_first_even* возвращает индекс найденного элемента.

При *n* = 1: происходит обращение к функции *index_last_odd(arr, len){}*. Функция получает на вход массив *arr* и значение переменной *len*. Аналогично функции *index_first_even* происходит перебор элементов массива, но в данном случае с конца. Поиск последнего нечётного элемента (остаток от деления на 2 равен 1) осуществляется так же с использованием функции *abs().*Функция *index_last_odd* возвращает индекс найденного элемента.

При n = 2: происходит обращение к функции *sum_between_even_odd (arr, len){}.* Функция получает на вход массив *arr* и значение переменной *len*. Объявляется целочисленная переменная *sum,* далее происходит перебор элементов массива посредствам цикла *for(){} (*Диапазон считается от первого чётного элемента (включая) и до последнего нечетного (не включая)). В переменную *sum* записывается сумма модулей элементов массива *arr* в данном диапазоне. Границы диапазона счёта определяют функции *index_first_even* и *index_last_odd*. Функция *sum_between_even_odd* возвращает значение переменной sum.

При *n* = 3: происходит обращение к функции *sum_before_even_and_after_odd (arr, len){}.* Функция получает на вход массив *arr* и значение переменной *len*. Объявляется целочисленная переменная *summ,* далее происходит перебор элементов массива посредствам цикла *for(){}* (Диапазон захватывает весь массив). В переменную *summ* записывается сумма модулей элементов массива *arr* в данном диапазоне. После этого, из переменной *summ*

вычитается значение, возвращённое функцией *sum_between_even_odd*, для получения суммы модулей элементов в диапазоне, данном в задании (От начала до первого чётного элемента (не включая) и от последнего нечётного элемента (включая) до конца массива).Функция *sum_before_even_and_after_odd* возвращает значение переменной *summ*.

Возвращаясь к функции *main(){},* точнее оператору *switch(){}* - при вводе определённого значения *n* оператор обращается к соответствующей функции и, с помощью функции *printf(),* выводит возвращённое значение (или сообщении о некорректных данных, при наличии таких), для выхода из оператора реализуется *break*.

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 2 3 4 5 6 7 8 9 10\n	0	Ответ верный.
2.	1 1 2 3 4 5 6 7 8 9 10\n	8	Ответ верный.
3.	2 1 2 3 4 5 6 7 8 9 10\n	35	Ответ верный.
4.	3 1 2 3 4 5 6 7 8 9 10\n	10	Ответ верный.
5.	8 1 2 3 4 5 6 7 8 9 10\n	данные некорректны	Ответ верный.
6.	1 -1 -1 -1 -1 -1 -1 -3 -4\n	6	Ответ верный.
7.	3 -1 -2 -3 -4 -5 -1 -2 -3\n	4	Ответ верный.
8.	2 -34 5 78 -9 0 -32 1 4 15 -3 -3 8 9 34\n	192	Ответ верный.
9.	3 5 7 9 33 4 77 121 67 -86 -15 76 0 32 -78 36 42 17 26 -17 -90 -9 16 28 14\n	121	Ответ верный.
10.	1 0 0 0 0 0 0 0 0 0 -2 -4 -6 7 0 0 0\n	12	Ответ верный.

Выводы.

Были изучены базовые управляющие конструкции языка Си.

Разработана программа, выполняющая считывание с клавиатуры исходных данных с помощью функции *scanf()* и цикла *while(){}*, для обработки команд пользователя использовались: условный оператор *if*, циклы *for(){}* и *while(){}* , оператор множественного выбора *switch(){},* функция *abs(),* функции, обрабатывающие входные данные, функция *printf().*

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <stdlib.h>
#define N 100
int index_first_even(int arr[], int len){
    for (int i=0; i<len; i++){
        if((abs(arr[i]))%2==0){
            return i;
        }
    }
}
int index_last_odd(int arr[], int len ){
    len = len -1;
    for (int i= len; i>=0; i--){
        if((abs(arr[i]))%2==1){
            return i;
        }
    }
}
int sum_between_even_odd(int arr[], int len){
    int sum=0;
    for (int i=index_first_even(arr, len); i<index_last_odd(arr, len); i++){
        sum=sum+abs(arr[i]);
    }
    return sum;
}
int sum_before_even_and_after_odd(int arr[], int len){
    int summ=0;
    for (int i = 0; i < len; i++){
    summ=summ+abs(arr[i]);
    }
    summ=summ-sum_between_even_odd(arr, len);
    return summ;
}
int main(){
    int n;
    scanf("%d", &n);
    int arr[N];
    int len=0;
    int i=0;
    char gap= ' ';
```

```
    while(i<N && gap==' '){
        scanf("%i%c", &arr[i], &gap);
        len=len+1;
        i=i+1;
    }
    switch (n){
        case 0:
            printf("%d\n",index_first_even(arr,len));
            break;
        case 1:
            printf("%d\n",index_last_odd(arr,len) );
            break;
        case 2:
            printf("%d\n",sum_between_even_odd(arr,len));
            break;
        case 3:
            printf("%d\n",sum_before_even_and_after_odd(arr,len));
            break;
        default:
            printf("Данные некорректны\n");
            break;
    }
    return 0;
}
```