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: Управляющие конструкции языка Си

Студент гр. 1304	Климов Г.А.	
Преподаватель	Чайка К.В.	

Санкт-Петербург
2021

Цель работы.

Исследование управляющих конструкция си, изучение основных способом написания программы и начало изучения функционального программирования

Задание.

Вариант №4.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.

В переменную *n* вводится первое значение с клавиатуры.Переменная *arr* -массив, который также вводится с клавиатуры, так как в задаче указан максимальный размер массива(100), то для размера массива я использовал пременную *N*, которая равна 100.Переменная *ch* - переменная типа char, нужна чтобы распознать символ перевода строки и остановить ввод.*i* - переменная для циклов, нужна чтобы работать с разными ячейками массива.Переменная *index* в функциях равна индексу искомого элемента.Переменная *sum* в функциях равна сумме модулей элементов массива в заданных переделах.Функция *int index_first_even(int arr[])* принимает массив и возвращает индекс первого чётного элемента.Функция *int index_last_odd(int arr[])* также получает аргументом массив и возвращает индекс последнего нечётного элемента.Функция *int sum_between_eve_odd(int arr[])* тоже получает массив на вход и находит сумму модулей от первого чётного элемента и до последнего нечётного, включая первый и не включая последний.Функция *int sum_before_even_and_after_odd(int arr[])* сумму модулей элементов массива, расположенных до первого чётного элемента (не включая элемент) и после последнего нечётного (включая элемент).

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 -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	0	Так как сначала вводится 0, то нужно вывести индекс первого чётного элемента, тоесть индекс -8, следовательно ответ 0.
2	6 23 43 12 5 6 7 8	“Данные некорректны”	Так как сначала была введена 6, это нарушает условия, а потому ответ “Данные некорректны”.
3	2 1 1 3 4 5 –6 7 8	15	Так как сначала ввели 2, то нужна сумма модулей от первого чётного(включая) до последнего нечётного(исключая), значит ответ 15.

Выводы.

Были изучены основные управляющие конструкции языка: условия, циклы, оператор switch.

Разработана программа, выполняющая считывание с клавиатуры исходных данных и команды пользователя. Для обработки команд пользователя использовался условный оператор *if*, цикл *for* и собственные функции, описаные в задании и реализованные через цикл *for* и вызовом внутри себя других функции, для избегания повторения кода.Во избежание возникновения исключительных ситуаций для нахождения модуля была использована стандартная функция *abs()*

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb1.c

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

int index_first_even(int arr[]){
```

```
	int index =0;
	int N=100;
	for(int i=0;i<N;i++){
		if (arr[i] % 2 == 0){
			index=i;
			break;
		}
	}
	return(index);
}

int index_last_odd(int arr[]){
	int N=100;
	int index =0;
	for(int i=0;i<N;i++){
		if (arr[i] % 2 != 0){
			index=i;
		}
	}
	return(index);
}

int sum_between_even_odd(int arr[]){
	int N=100;
	int sum =0;
	for(int i=index_first_even(arr);i<index_last_odd(arr);i++){
		sum +=abs(arr[i]);
	}
	return(sum);
}

int sum_before_even_and_after_odd( int arr[]){
	int N=100;
	int sum =0;
	for(int i=0;i<index_first_even(arr);i++){
		sum +=abs(arr[i]);
```

```
        }

        for(int i=index_last_odd(arr);i<N;i++){
                sum +=abs(arr[i]);
        }

        return(sum);
}


int main(){
int n;
int N=100;
scanf("%d",&n);
int arr[N];
char ch;
int i=0;
for(i=0;i<N;i++){
        arr[i]=0;
 }
 i=0;

 for(i=0;i<N;i++){
        scanf("%d%c",&arr[i],&ch);
        if(ch =='\n'){
                break;
        }
 }

switch(n){
        case 0:
                printf("%d\n",index_first_even(arr));
                break;
        case 1:
                printf("%d\n",index_last_odd(arr));
                break;
```

```
        case 2:
                printf("%d\n",sum_between_even_odd(arr));
                break;
        case 3:
                        printf("%d\n",sum_before_even_and_after_odd(arr));
                break;
        default:
            printf("Данные некорректны");
}



return 0;



}
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Таблица 2 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
4	1 3 9 5 2 –5 –6 –7 8	6	Так как введена 1, то нужно вывести индекс последнего нечётного(-7), то есть 6.
5	3 1 1 2 99 –33 6 6	47	Так как введена 3, то нужно вывести сумму модулей до первого чётного(исключая) и после последнего нечётного(включая), значит ответ 47.