Remove Watermark Now

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

pdfelement

Студент гр. 0382 _____ Довченко М.К.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2020

Remove Watermark Now

Цель работы.

Изучение базовых управляющих конструкций языки 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. (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".

pdfelement

Основные теоретические положения.

В данной работе была использована функция abs() из библиотеки stdlib.h для нахождения модуля числа. Также были использованы функции scanf() и printf() для ввода и вывода из библиотеки stdio.h. Кроме этого были использованы операторы if(){} else{}, for (){}, while(){}, switch(){}

Remove Watermark Now

Выполнение работы.

В функции *main{}* объявляется целочисленная переменная val, которой с помощью функции *scanf()* присваивается целочисленное значение. Далее объ является целочисленный массив *array* размером 100 и целочисленная переме нная *size* равная 0, которая показывает количество элементов в массиве. В сл едующей строчке объявляется символьная переменная spaceb = ‘ ‘. Далее в теле цикла *while (size <100 && spaceb==’ ‘){}* применяется функция *scanf()*, с помо щью которой вводится целый элемент массива *array[]* с индексом *size++* и сим вол spaceb. Далее применяется оператор *switch(val){}*, который в зависимости о т значения val, будет выполнять различные команды.

Если val равняется 0, то с помощью функции *printf()* печатается значение функции *index_first_even(array, size)*. Функция *index_first_even(int[], int)* получа ет на вход целочисленный массив array и целое число size, затем в функции соз даётся локальная целочисленная переменная counter0, равная 0. Используя цикл for(counter0 = 0; counter0 <= size; counter0++), в теле которого counter0 увеличи вается на 1 за итерацию, удаётся найти индекс первого чётного элемента(фукнц ия *abs()* используется, так как если *array[counter0]* будет отрицательным, то в с лучае нечётности элемента значение array*[counter0]%2* будет равно -1 и цикл завершится. Функция возвращает значение counter0. Для выхода из оператора *switch* используется *break*.

Если val равняется 1, то с помощью функции *printf()* печатается значение функции *index_last_odd(array, size)*. Функция *index_last_odd(int [], int){}* получа ет на вход целочисленный массив array и целое число size. Затем, используя цик л for(counter1 = size-1; counter1 >= 0; counter1--), в теле которого counter1 умен ьшается на 1 за итерацию, удаётся найти индекс последнего нечётного элем ента в массиве. Функция возвращает значение counter1. Для выхода из операто ра *switch* используется *break*.

pdfelement

Remove Watermark Now

Если val равняется 2, то с помощью функции *printf()* печатается значение функции *sum_between_even_odd(array, size)*. Функция *sum_between_even_odd (int [], int){}* получает на вход целочисленный массив array и целое число size. Объ является 2 локальные целочисленные переменные: counter 2, sbed = 0, где counter2 присваевается индекс первого чётного элемента массива(находится с п омощью ранее описанной функции), sbed – искомая сумма. Далее с помощью цикла *for(counter2 = index_first_even(array, size); counter2 < index_last_odd(array , size); counter2++){}* с телом *sbed += abs(array[counter2])*, находится сумма ч ленов массива от первого чётного(включая) до последнего нечетного(исключ ая). Функция возвращает значение *summ*. Для выхода из оператора switch исп ользуется *break*.

Если val равняется 3, то с помощью функции *printf()* печатается значение функции *sum_before_even_and_after_odd(array, size)*. Функция *sum_before_even_and_after_odd (int [], int){}* получает на вход целочисленный массив и целое число. Объявляются три целочисленные локальные переменн ые *sbeaao=0*, где *sbeaao* – искомая сумма, *counter3 и counter4*. Далее с помощью 2х циклов находится сумма всех элементов массива. После чего функция возв ращает значение *sbeaao*. Для выхода из оператора switch(){} используется *break*.

При значении val, отличном от 0,1,2 или 3, с помощью функции *printf*() пе чатается строка “Данные некорректны”. Для выхода из оператора *switch* (){} используется *break*.

pdfelement

Remove Watermark Now

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\n	0	Программа работает правильно.
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 правильно.
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 правильно.
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 правильно.
5	0 8 -23 -30 -11 -28 15	0	Программа работает правильно.
6	1 8 -23 -30 -11 -28 15	5	Программа работает правильно.
7	2 8 -23 -30 -11 -28 15	100	Программа работает правильно.
8	3 8 -23 -30 -11 -28 15	15	Программа работает правильно.

pdfelement

Remove Watermark Now

Выводы.

В ходе работы были изучены основные управляющие конструкции языка Си.

Разработана программа, выполняющая считывание с клавиатуры исходных данных с помощью функции *scanf()* и цикла *while(){}* и команды пользователя, для обработки команд пользователя использовалась символьная переменная *sym*, хранящая код символа ’ ‘, написаны функции, обрабатывающие входные данные, описание функций приведено в блоке “Выполнение работы”. С помощью оператора *switch(){}* и функции *printf()* реализован вывод значения определенной функции в зависимости от значения переменной *k*.

pdfelement

Remove Watermark Now

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lbfinal.c

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

int index_first_even(int[], int);
int index_last_odd(int[], int);
int sum_between_even_odd(int[], int);
int sum_before_even_and_after_odd(int[], int);

int main(){
    int size = 0, array[100], val;
    char spaceb = ' ';

    scanf("%d", &val);

    while(size < 100 && spaceb == ' '){
        scanf("%d%c",&array[size++], &spaceb);
    }

    switch (val){
        case 0:
            printf("%d\n", index_first_even(array,
size));
            break;
        case 1:
            printf("%d\n", index_last_odd(array, size
));
            break;
        case 2:
            printf("%d\n", sum_between_even_odd(array
, size));
            break;
        case 3:
            printf("%d\n", sum_before_even_and_after_
odd(array, size));
            break;
        default:
            printf("Данные некорректны\n");
            break;
    }
    return 0;
}
```

pdfelement

Remove Watermark Now

```
    int index_first_even(int array[], int size){
        int counter0;
        for(counter0 = 0; counter0 <= size; counter0++)
            if(abs(array[counter0])%2 == 0)
                return counter0;
    }

    int index_last_odd(int array[], int size){
        int counter1;
        for(counter1 = size-1; counter1 >= 0; counter1--)
            if(abs(array[counter1])%2 == 1)
                return counter1;
    }

    int sum_between_even_odd(int array[], int size){
        int counter2, sbed = 0;
        for(counter2 = index_first_even(array, size); counter2 <
    index_last_odd(array, size); counter2++)
            sbed += abs(array[counter2]);
        return sbed;
    }

}   int sum_before_even_and_after_odd(int array[], int size){
        int counter3, counter4, sbeaao = 0;
        for(counter3 = index_first_even(array, size) - 1;  counter3 >
    = 0;  counter3--)
            sbeaao += abs(array[counter3]);
        for(counter4 = index_last_odd(array, size); counter4 < size;
    counter4++)
            sbeaao += abs(array[counter4]);
        return sbeaao;
    }
```

pdfelement