МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Программирование»

Тема: Условия, циклы, оператор switch

Студент гр. 1304 _____ Нго Тхи Йен

Преподаватель _____ Чайка К.В

Санкт-Петербург

2021

Цель работы.

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 : максимальное по модулю число в массиве. (abs_max)

1 : минимальное по модулю число в массиве. (abs_min)

2 : разницу между максимальным по модулю и минимальным по модулю элементом. (diff)

3 : сумму элементов массива, расположенных после максимального по модулю элемента (включая этот элемент). (sum)

иначе необходимо вывести строку "Данные некорректны".

Дополнительная информация.

Операторный блок - несколько операторов, сгруппированные в единый блок с

помощью фигурных скобок.

{ [<оператор 1>...<оператор N>] }

Условный оператор

if (<выражение>) <оператор 1> [else <оператор 2>]

Если выражение интерпретируется как истина, то оператор1 выполняется. Может иметь необязательную ветку else, путь выполнения программы пойдет в случае если выражение ложно. В языке С любое ненулевое выражение расценивается как истина.

Оператор множественного выбора

switch (<выражение>)

{ case <константное выражение 1>: <операторы 1>

...

case <константное выражение N>: <операторы N>

[default: <операторы>]

}

Выполняет поочередное сравнение выражения со списком константных выражений. При совпадении, выполнение программы начинается с соответствующего оператора. В случае, если совпадений не было, выполняется необязательная ветка default. Важно помнить, что операторы после первого совпадения будут выполняться далее один за другим. Чтобы этого избежать, следует использовать оператор break.

Цикл с предусловием

while (<выражение>) <оператор>

На каждой итерации цикла происходит вычисление выражения и если оно истинно, то выполняется тело цикла

Цикл с постусловием

do <оператор> while <выражение>;

На каждой итерации цикла сначала выполняется тело цикла, а после вычисляется выражение. Если оно истинно — выполняется следующая итерация.

Цикл со счетчиком

for ([<начальное выражение>]; [<условное выражение>]; [<выражение приращения>])

<оператор>

Условием продолжения цикла как и в цикле с предусловием является некоторое

выражение, однако в цикле со счетчиком есть еще 2 блока — начальное

выражение, выполняемое один раз перед первым началом цикла и выражение

приращения, выполняемое после каждой итерации цикла. Любая из трех частей

оператора for может быть опущена.

Оператор break — досрочно прерывает выполнение цикла.

Оператор continue — досрочный переход к следующей итерации цикла.

Экспериментальные результаты.

Создание проекта

1. Создание abs_max

```
int abs_max(int arr[], int size)
{
    int max = abs(arr[0]);
    int result = arr[0];
    for(int i = 1; i < size; i++){
        if (max < abs(arr[i]))
        {
            max = abs(arr[i]);
            result = arr[i];
        }
    }
    return result;
}
```

2. Создание abs_min

```
int abs_min(int arr[], int size)
{
    int min = abs(arr[0]);
    int result = arr[0];
    for(int i = 1; i < size; i++){
       if(min > abs(arr[i]))
       {
            min = abs(arr[i]);
            result = arr[i];
       }
    }
    return result;
}
```

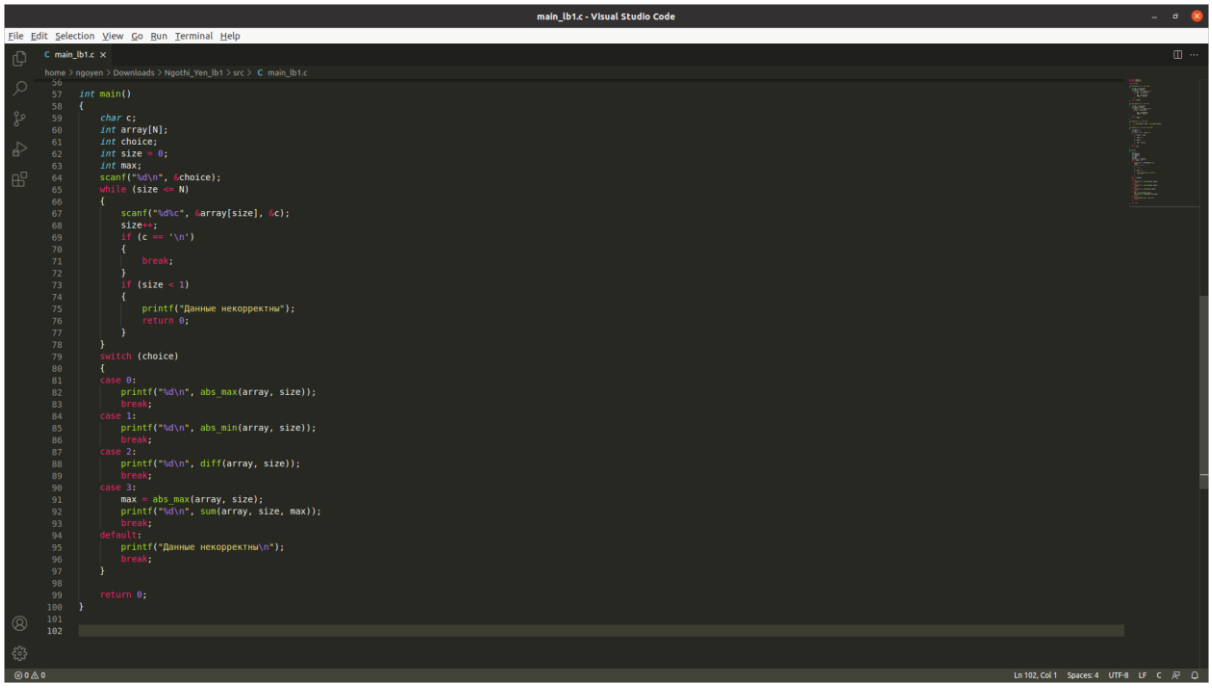
3. Создание diff

```
int diff(int arr[], int size)
{
    return(abs_max(arr, size) - abs_min(arr,size));
}
```

4. Создание sum

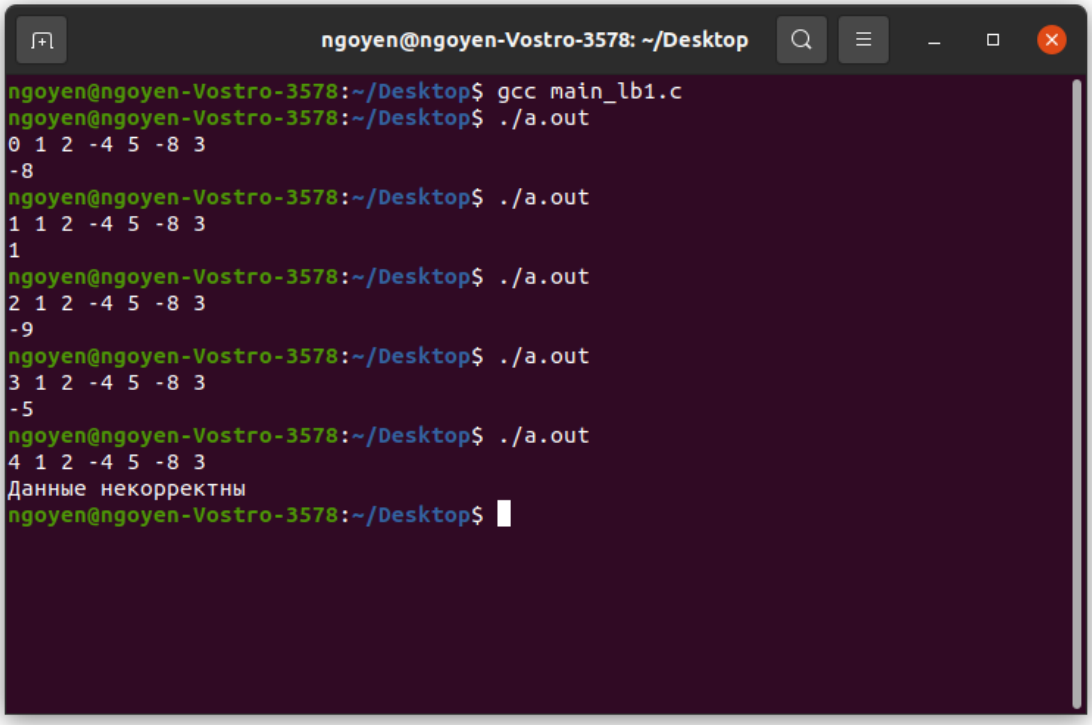
```c
int diff(int arr[], int size)
{
    return(abs_max(arr, size) - abs_min(arr,size));
}

int sum(int arr[], int size, int max)
{
    int start = 0;
    int sum = 0;
    for (int i = 0; i < size; i++)
    {
        if (arr[i] == max)
        {
            start = 1;
        }
        if (start)
        {
            sum += arr[i];
        }
    }
    return sum;
}
```

5. Создание main_lb1.c



Выводы.

Результатом лабораторной работы является полностью рабочая программа, удовлетворяющая всем указанным в цели лабораторной работы условиям. Проверим, что программа работает правильно.


ngoyen@ngoyen-Vostro-3578: ~/Desktop
ngoyen@ngoyen-Vostro-3578:~/Desktop$ gcc main_lb1.c
ngoyen@ngoyen-Vostro-3578:~/Desktop$ ./a.out
0 1 2 -4 5 -8 3
-8
ngoyen@ngoyen-Vostro-3578:~/Desktop$ ./a.out
1 1 2 -4 5 -8 3
1
ngoyen@ngoyen-Vostro-3578:~/Desktop$ ./a.out
2 1 2 -4 5 -8 3
-9
ngoyen@ngoyen-Vostro-3578:~/Desktop$ ./a.out
3 1 2 -4 5 -8 3
-5
ngoyen@ngoyen-Vostro-3578:~/Desktop$ ./a.out
4 1 2 -4 5 -8 3
Данные некорректны
ngoyen@ngoyen-Vostro-3578:~/Desktop$