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

ОТЧЕТ

по лабораторной работе№2

по дисциплине «Программирование»

ТЕМА: "Условия, циклы, оператор switch"

Студент гр. 7381 _____ Смирнов М.А.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2017

Цель работы:

Познакомиться с оператором *switch*, с циклами *for, while,do while* и с операторами *case, break, default.*

В текущей директории создать проект с *make*-файлом. Главная цель должна приводить к сборке проекта. Файл, который реализует главную функцию, должен называться *menu.c*; исполняемый файл - *menu*. Определение каждой функции должно быть расположено в отдельном файле, название файлов указано в скобках около описания каждой функции.

Реализовать функцию-меню, 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 нулевого элемента. (index_first_zero.c)
1 : индекс последнего нулевого элемента. (index_last_zero.c)
2 : Найти сумму модулей элементов массива, расположенных от первого нулевого элемента и до последнего. (sum_between.c)
3 : Найти сумму модулей элементов массива, расположенных до первого нулевого элемента и после последнего.  (sum_before_and_after.c);

иначе необходимо вывести строку "Данные некорректны".
```

Основные теоретические положения

Заголовочные файлы, необходимые для создания проекта:

1. - содержит прототипы функций "*int printf(const char* format [, argument]...);*" и "*int scanf(const char* format [, argument]...);*", которые используются для ввода из потока ввода и вывода в поток вывода.

Синтаксис: *#include < stdio.h > int printf (const char *format, ...);*

Аргументы: *format* – указатель на строку с описанием формата.
Возвращаемое значение: При успешном завершении вывода возвращается количество выведенных символов. При ошибке возвращается отрицательное число.

Описание: Функция *printf* выводит в стандартный поток вывода строку отформатированную в соответствии с правилами, указанными в строке, на которую указывает аргумент *format*.
Правила задаются набором трех типов директив:
1. Обычные символы (кроме '%' и '\'), которые выводятся без изменения;
2. Спецификаторы формата;
3. Специальные сиволы.

Синтаксис: *#include < stdio.h > int scanf(const char *format, ...);*

Аргументы: *format* – указатель на строку с описанием формата.

Возвращаемое значение: Возвращает число, равное количеству полей, значения которых были действительно присвоены переменным. Если до присвоения значения первого поля произошла ошибка, возвращается EOF.

Описание: Функция *scanf()* является процедурой ввода общего назначения, считывающей данные из потока *stdin*.

2. - содержит прототип функции "*int abs(int n);*", возвращающей абсолютное значение числа.

Синтаксис: *#include < string.h > int abs(int n);*

Аргументы: n – целое значение.

Возвращаемое значение: Возвращает абсолютное значение числа.

Исходный код проекта:

1.menu.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "index_first_zero.h"
#include "index_last_zero.h"
#include "sum_between.h"
#include "sum_before_and_after.h"

int main()
{
int vybor;
int mas[100];
char c;
int i=0;
scanf("%d%c",&vybor,&c);
while (c!='\n')
{
    scanf("%d%c",&mas[i],&c);
    i++;
}
i--;
switch(vybor)
{
    case 0:
        printf("%d\n",index_first_zero(mas,i));
        break;
    case 1:
        printf ("%d\n",index_last_zero(mas,i));
        break;
    case 2:
        printf ("%d\n",sum_between(mas,i));
        break;
    case 3:
        printf ("%d\n",sum_before_and_after(mas,i));
        break;
    default:
        printf ("Данные некорректны\n");
}

    return 0;
```

2.index_first_zero.c

```c
#include <stdio.h>

int index_first_zero(int mas[],int i)
{

int pierre;
for(pierre=0;pierre<=i;++pierre)
{
    if(mas[pierre]==0)
        break;
}


        return pierre;
}
```

3.index_first_zero.h

```c

#pragma once

int index_first_zero(int mas[],int i);

```

4.index_last_zero.c

```c
#include <stdio.h>

int index_last_zero(int mas[],int i)
{

int bolk;
for(bolk=i;bolk>=0;bolk--)
{
 if(mas[bolk]==0)
    break;


}

        return bolk;

}
```

5. index_last_zero.h

```c

#pragma once

int index_last_zero(int mas[],int i);

```

6. sum_between.c

```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include "index_first_zero.h"
#include "index_last_zero.h"

int sum_between(int mas[],int i)
{
    int beatsum=0,zero;
    int fir = index_first_zero(mas,i);
    int las = index_last_zero(mas,i);
    for(zero=fir;zero<las;++zero)
    {
        beatsum=beatsum+abs(mas[zero]);
    }
    return beatsum;
}
```

7. sum_between.h

```c
#pragma once
int sum_between(int mas[],int i);
```

8. sum_before_and_after.c

```c
#include <stdio.h>
#include <stdlib.h>
#include "index_first_zero.h"
#include "index_last_zero.h"

int sum_before_and_after(int mas[],int i)
{
    int a=0,b=0,zero,sub;
    int fir = index_first_zero(mas,i);
    int las = index_last_zero(mas,i);

    for(zero=0;zero<fir;++zero)
    {
        a+=abs(mas[zero]);
    }
    zero=0;
    for(zero=las;zero<=i;++zero)
    {
        b=b+abs(mas[zero]);
    }
    sub=a+b;

    return sub;
}
```

9. sum_before_and_after.h

```c

#pragma once

int sum_before_and_after(int mas[],int i);

```

10. Makefile

```
menu: menu.o index_first_zero.o index_last_zero.o sum_between.o sum_before_and_after.o
	gcc menu.o index_first_zero.o index_last_zero.o sum_between.o sum_before_and_after.o -o menu
menu.o: menu.c
	gcc -c menu.c
index_first_zero.o: index_first_zero.c index_first_zero.h
	gcc -c index_first_zero.c
index_last_zero.o: index_last_zero.c index_last_zero.h
	gcc -c index_last_zero.c
sum_between.o: sum_between.c sum_between.h
	gcc -c sum_between.c
sum_before_and_after.o: sum_before_and_after.c sum_before_and_after.h
	gcc -c sum_before_and_after.c
```

Вывод: В ходе лабораторной работы была получена работающая программа согласно указаниям. В работе существует вариативность выполнения действий таких как: нахождение индекса 1-ого/последнего нулевого элемента, сумма элементов массива и т.п. В процессе написания кода, были получены новые знания о работе циклов, операторов, массивов в языке Си.