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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Программирование»

Тема: «Условия, циклы, оператор switch»

Студент гр. 7381 Лауцюс М.

Преподаватель Берленко Т.А.

Санкт-Петербург

2017

Цель работы.

Ознакомление с оператором выбора switch, с циклами for, while , do while, с массивами, с операторами break и return, а также с функциями стандартной библиотеки для ввода/вывода.

Задание

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. Определение каждой функции должно быть расположено в отдельном файле**,** название файлов указано в скобках около описания каждой функции.

Должна быть реализована функция-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 стандартной библиотеки языка Си, используемые в данной лабораторной работе:

1. stdio.h - содержит определения макросов, константы и объявления функций и типов, используемых для различных операций стандартного ввода и вывода.

Она предоставляет доступ к использованию функий:
int scanf(const char* format [, argument]…), int printf (const char* format [,argument]…), int getc (FILE * filestream) и int ungetc(int character, FILE * filestream).

int getc (FILE * filestream):

Описание:

Функция возвращает символ из потока “filestream”, на который ссылается внутренний индикатор позиции файла. Внутренний индикатор позиции в файле, после срабатывания этой функции сдвигается на один символ, таким образом, теперь он указывает на следующий символ.

Параметры:

“filestream”

Указатель на объект типа FILE, который идентифицирует поток, для выполнения с ним различных операций.

Возвращаемое значение:

Считанный символ возвращается в виде целого значения.

Если конец файла достигнут или в процессе чтения происходит ошибка, функция возвращает EOF и соответствующие индикаторы ошибки или конца файла устанавливаются.

int ungetc(int character, FILE * filestream):

Описание:

Функция ungetc возвращает только что прочитанный символ обратно в поток ввода “filestream”, через параметр “character”. Внутренний индикатор позиции файла уменьшается обратно, на предыдущее положение, так что этот символ возвращается при следующем вызове операции чтения для этого потока.

Параметр “character” может содержать любой символ, например, последний символ прочитанный из потока в предыдущей операции или любой другой. В обоих случаях, значение, полученное по следующей операции чтения является значением функции ungetc, независимо от символа “character”.

Параметры:

“character”

Символ, возвращаемый обратно в поток. Символ передается, как значение типа int.

“filestream”

Указатель на объект типа FILE, который идентифицирует входной поток.

Возвращаемое значение:

В случае успеха, возвращается целочисленное значение символа, который был перенесен обратно в поток. В противном случае, возвращается значение EOF, и поток остается неизменным.

Вывод:

Были получены знания о массивах в языке Си: использование, расположении в памяти, и т. д. Был освоен оператор выбора switch: синтаксис, применение, операторы case, break и default. В лабораторной работе был также использован цикл for.

Исходный код проекта:

Файл Makefile

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

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

Файл menu.c

```
#define N 100
#include "index_first_zero.h"
#include "index_last_zero.h"
#include "sum_between.h"
#include "sum_before_and_after.h"
#include <stdio.h>
int read(int* mass){
    int c, size=0;
    while ((c = getc(stdin)) != '\n' && size<N) {
        ungetc(c, stdin);
        scanf("%d", &mass[size++]);
    }
    return size;
}
int main() {
    int mass[N];
    int size;
    int val;
    scanf("%d", &val);
    size=read(mass);
    switch (val) {
    case 0:
        printf("%d", index_first_zero(mass, size));
```

```
        break;
    case 1:
        printf("%d", index_last_zero(mass, size));
        break;
    case 2:
        printf("%d", sum_between(mass, size));
        break;
    case 3:
        printf("%d", sum_before_and_after(mass, size));
        break;
    default:
        printf("Данные некорректны");
    }
    printf("\n");
    return 0;
}
```

Файл index_first_zero.c

```
int index_first_zero(int* mass, int size) {
    int i;
    for (i = 0; i < size; i++)
        if (mass[i] == 0)
            return i;
}
```

Файл index_first_zero.h

```
#pragma once
int index_first_zero(int [], int);
```

Файл index_last_zero.c

```
int index_last_zero(int mass[], int size) {
    int i;
```

```
    for (i = size-1; i>=0; i--)
        if (mass[i] == 0)
            return i;
}
```

Файл sum_before_and_after.c

```
#pragma once
int index_last_zero(int*, int);
```

Файл index_last_zero.c

```
#include "index_first_zero.h"
#include "index_last_zero.h"

int sum_before_and_after(int* mass, int size){
    int first = index_first_zero(mass, size),
        last = index_last_zero(mass, size),
        result = 0,
        i = 0;
    for (; i < size; i++){
        if(i == first)
            i = 1+last;
        result += (mass[i] > 0 ? mass[i] : -mass[i]);
        }
    return result;
}
```

Файл sum_before_and_after.h

```
#pragma once
int sum_before_and_after(int [], int);
```

Файл sum_between.c

```
#include "index_first_zero.h"
```

```
#include "index_last_zero.h"

int sum_between(int mass[], int size) {
    int first = index_first_zero(mass, size),
        last = index_last_zero(mass, size),
        result = 0,
        i = first;
    for (; i < last; i++)
        result += (mass[i]>0 ? mass[i] : -mass[i]);
    return result;
}
```

Файл sum_between.h

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