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 _____ Трушников А.П.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2017

Цель работы.

Познакомиться с массивами; познакомиться с оператором выбора switch; познакомиться с циклами for (;;), while (), do while (); познакомиться с операторами break и return; 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 чётного элемента. (index_first_even.c)

1: индекс последнего нечётного элемента. (index_last_odd.c)

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.c)

3: Найти сумму модулей элементов массива, расположенных до первого чётного элемента (не включая элемент) и после последнего нечетного (включая элемент). (sum_before_even_and_after_odd.c)

Иначе необходимо вывести строку “Данные некорректны”.

Основные теоретические положения.

Заголовочные файлы стандартной библиотеки языка С, необходимые для выполнения данной лабораторной работы:

1. Функция «int main()». В этой функции заголовочный файл «stdio.h» (стандартный заголовочный файл ввода-вывода) подключает прототип функций «int scanf(const char* format [, argument]…);», «int printf (const char* format [,

argument]…);», которые используются для ввода из потока ввода и вывода в поток вывода. А так же прототипы функций «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. Вы можете использовать любую функцию ferror или feof чтобы определить, произошла ошибка или был достигнут конец файла.

«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.

2. В функциях «sum_between_even_odd(int *a, int len);» и « sum_before_even_and_after_odd(int *a, int len);». В них заголовочный файл «stdlib.h» подключает прототип функции «int abs(int n);».

«int abs(int n)»:

Описание:

Функция вычисляет абсолютную величину (модуль) значения, передаваемого в качестве аргумента через параметр n.

Параметры:

- n

Целое значение.

Возвращаемое значение:

Модуль числа n.

Вывод:

Познакомился с массивами в C: синтаксисом, использованием, расположением в памяти, и т. д. Познакомился с оператором выбора switch: синтаксисом, использованием, операторами case, break и default. Познакомился с циклами for (;;), while (), do while (): синтаксисом, использованием, операторами break для выхода из цикла.

Исходный код проекта:

Файл “Makefile”

```
all: menu

menu:menu.o index_first_even.o index_last_odd.o sum_between_even_odd.o
sum_before_even_and_after_odd.o
	gcc menu.o index_first_even.o index_last_odd.o  sum_between_even_odd.o
sum_before_even_and_after_odd.o -o menu

main.o:menu.c
	gcc -c menu.c


index_first_even.o:index_first_even.c index_first_even.h
	gcc -c index_first_even.c

index_last_odd.o:index_last_odd.c index_last_odd.h
	gcc -c index_last_odd.c

sum_between_even_odd.o:sum_between_even_odd.c sum_between_even_odd.h
	gcc  -c sum_between_even_odd.c

sum_before_even_and_after_odd.o:sum_before_even_and_after_odd.c
sum_before_even_and_after_odd.h
	gcc -c sum_before_even_and_after_odd.c
clean:
	rm -rf *.o hello
```

Файл “index_first_even.c”

```
#include <stdio.h>

int index_first_even(int *a, int len){

int i;
int min;

 for(i=0;i<len;i++) {
     if(a[i] % 2 ==0){
        min=i;

     break;}
}
```

```
return min;
}
```

Файл “index_first_even.h”

```
#pragma once
int index_first_even(int *a, int len);
```

Файл “index_last_odd.c”

```
#include <stdio.h>

int index_last_odd(int *a, int len){


int i;
int max;



  for(i=0;i<len;i++) {
    if(a[i] % 2 !=0) {

      max=i;
    }

 }

return max;

}
```

Файл “index_last_odd.h”

```
#pragma once
int index_last_odd(int *a, int len);
```

Файл “main.c”

```
#include "index_first_even.h"
#include "index_last_odd.h"
#include "sum_between_even_odd.h"
#include "sum_before_even_and_after_odd.h"
#include <stdio.h>


int main(){
int a[100] = {};
int i=0;
```

```
int c;
int var;
int number=0;


scanf("%d",&var);
 while( ((c=getchar())) !='\n'){
     ungetc(c,stdin);
     scanf("%d",&i);
    a[number++]=i;
    }

switch(var){
    case 0:
        printf("%d\n", index_first_even(a,number));
        break;

    case 1:
        printf("%d\n", index_last_odd(a,number));
        break;

    case 2:
        printf("%d\n", sum_between_even_odd(a,number));
        break;

    case 3:
        printf("%d\n", sum_before_even_and_after_odd(a,number));
        break;
    default:
        printf("Данные некорректны");
        break;


}


return 0;
}
```

Файл “sum_before_even_and_after_odd.c”

```
#include "index_first_even.h"
#include "index_last_odd.h"

#include <stdlib.h>
```

```
int sum_before_even_and_after_odd(int *a, int len){
   int i;
   int sum1=0,sum2=0,sum3;

   for(i=0;i<index_first_even(a,len);i++){
      sum1 +=abs(a[i]);

   }

   for(i=index_last_odd(a,len);i<=len;i++){
      sum2 +=abs(a[i]);

   }
   sum3=sum1+sum2;
   return sum3;

}
```

Файл “sum_before_even_and_after_odd.h”

```
#pragma once
int sum_before_even_and_after_odd(int *a, int len);
```

Файл “sum_between_even_odd.c”

```
#include "index_first_even.h"
#include "index_last_odd.h"

#include <stdlib.h>

 int sum_between_even_odd(int *a, int len){
   int sum=0;
   int i;

   for(i=index_first_even(a,len);i<index_last_odd(a,len);i++){
      sum +=abs(a[i]);

   }
   return sum;

}
```

Файл “sum_between_even_odd.h”

```
#pragma once
int sum_between_even_odd(int *a, int len);
```