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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Программирование»

Тема: Сборка программ в Си

Студент гр. 0304 _____ Мажуга Д.Р

Преподаватель _____ Чайка К.В

Санкт-Петербург

2020

Цель работы.

Изучить процесс сборки программ на языке С, научиться создавать заголовочные файлы, разбивать проект на файлы и собирать его с помощью утилиты make.

Задание.

Вариант No1.

Реализуйте программу, на вход которой подается одно из значений 0, 1, 2, 3 и массив целых чисел размера не больше 20. Числа разделены пробелами.

Строка заканчивается символом перевода строки.

В зависимости от значения, функция должна выводить следующее:

0: индекс первого отрицательного элемента. (index_first_negative.c)

1: индекс последнего отрицательного элемента. (index_last_negative.c)

2: Найти произведение элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент). (multi_between_negative.c)

3: Найти произведение элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент). (multi_before_and_after_negative.c).

Иначе вывести строку “Данные некорректны”.

Основные теоретические положения.

Были использованы заговолочные файлы стандартных библиотек: stdio и stdlib (из данных библиотек были задействованы функции int scanf(const char*, ...) и int getchar(void)). Также были использованы управляющие конструкции языка Си: циклы с счетчиком for и с предусловием while, условный оператор if и оператор множественного выбора (ветвления) switch. Также для разбиения исходного кода на несколько файлов использовались расширения .c, созданы заголовочные файлы .h, использована утилита make и создан специальный Makefile, главная цель которого собирает программу с помощью утилиты gcc.

Выполнение работы.

1. Подключение стандартных библиотек stdio и stdlib.

2. При помощи директивы процессора #define задается символическая константа MAX, которая обозначает максимальное количество элементов массива.

3. Создаются прототипы функций, на которые в дальнейшем будет ссылаться оператор множественного выбора switch:

int index_first_negative (int s[], len)

int index_last_negative (int s[], len)

int multi_between_negative (int s[], len)

int multi_before_and_after_negative (int s[], len)

В качестве аргументов в этих функциях используются массив s [] и его длина len.

4. В функции int main () объявляется переменная n, которая отвечает за выбор используемой функции, создается массив arr[] типа int с размерностью MAX и количеством считанных элементов (длиной) i.

5. При помощи функции scanf вызывается переменная n, выбранная пользователем.

6. При помощи цикла с предусловием while и функции scanf происходит ввод и считывание элементов целочисленного массива. При этом в цикле while указано условие, когда считывание данных прекращается: превышение размерности массив (i<MAX) и символ переноса срока (getchar() != '\n' – если введенный символ совпадает с символом переноса строки, то считывание символов прекращается). Также после выполнения считывания символа происходит инкрементирование i – счётчика массива.

7. Далее при помощи оператора множественного выбора switch происходит вызов функции, соответствующей выбору пользователя. Результат выполнения функции выводится на экран.

8. Функции, используемые в данной работе, работают следующим

образом:

index_first_negative (int s [], int len)

1) объявляется переменная i типа int

2) при помощи цикла for переменная i инкрементируется с 0 и до момента нахождения первого отрицательного элемента последовательности

3) функция возвращает значение переменной i

index_last_negative (int s [], int len)

1) объявляется переменная i типа int

2) при помощи цикла for переменная i декрементируется от (len-1) до момента нахождения первого с конца отрицательного элемента последовательности

3) функция возвращает значение i

multi_between_negative (int s[], int len)

1) вводятся символические переменные i и k (порядковые номера первого и последнего отрицательных чисел соответственно), и переменная pr, которая отвечает за произведение элементов от первого отрицательного до числа, предшествующего последнему отрицательному

2) при помощи цикла for производится перемножение элементов массива, начиная с элемента с порядковым номером i и заканчивая элементом, расположенного перед элементом с порядковым номером k

3) функция возвращает значение pr

index_before_and_after_negative(int s[], int len)

1) вводятся символические переменные i и k (порядковые номера первого и последнего отрицательных чисел соответственно), переменная a, которая отвечает за порядковый номер элемента, и переменная pr, которая отвечает за произведение элементов от 0 до элемента, предшествующего первому отрицательному элементу, и от последнего отрицательного элемента до последнего элемента массива

2) при помощи двух циклов for производится подсчет произведений. Первый цикл for производит перемножение элементов массива от s [0] до s [i-1], а второй цикл for производит перемножение элементов от s [k] до arr [len]

3) функция возвращает значение pr

2. Создание заговолочных файлов, которые содержат прототипы функций:

index_first_negative.h:

int index_first_negative (s [], int len);

index_last_negative.h:

int index_last_negative (s [], int len);

multi_between_negative.h:

int multi_between_negative (s [], int len);

multi_before_and_after_negative.h:

int multi_before_and_after_negative (s [], int len);

3. Создание файла menu.c, в котором заключается функция int main() с подключением заголовочных файлов из пункта 2:

#include “index_first_negative.h”

#include “index_last_negative.h”

#include “multi_between_negative.h”

#include “multi_before_and_after_negative.h”

4. Создание файлов index_first_negative.c, index_last_negative.c, multi_between_negative.c и multi_before_and_after_negative.c , которые соответствуют исходным функциям

5. Создание Makefile с целью all, которая представляет собой вызов gcc с параметрами в виде всех выше указанных файлов с расширением .c. Параметр -o menu задаёт имя menu итоговому исполняемому файлу.

Исходный код программы находится в приложении А

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

No п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 -1 2 -2 3 -3 4 -4 8	1	OK
2.	1 10 -1323 3231 -124 34232 -35 989 -567 5678 -8094 3547 -676	11	OK
3.	2 15 48 -547 544 -897 51455 -485 -784 55147 123 -5689 5584 5414	980857344	OK
4.	3 23 45 56 67 -45 565 32435 6898 436531 -565 45 123 89	-1835211304	OK
5.	4 2394 33 -34 546 -67 -789 634	Данные некорректны	OK

Выводы.

Был изучен процесс сборки программ в C с помощью утилиты make.

Разработана программа, выполняющая считывание с клавиатуры исходных данных со стандартного потока вывода и выбор пользователя, которая затем выводит полученную в результат, получаемый при выполнении программы. После этого все пронумерованные операции были внесены в отдельные файлы index_first_negative.c, index_last_negative.c, multi_between_negative.c, multi_before_and_after_negative.c. Также были созданы заголовочные файлы, эквивалентные данным программам, а общий вызов необходимой операции производился с помощью файла menu.c. Сборка производилась при помощи утилиты make.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

***i**ndex_first_negative.h:*

```
int index_first_negative(int s[], int len);
```

index_first_negative.c:

```
#include "index_first_negative.h"

int index_first_negative(int s[], int len)
{
   int i;
   for(i = 0; i < len; i++)
      if(s[i] < 0)
         break;
   return i;
}
```

index_last_negative.h:

```
int index_last_negative(int s[], int len);
```

index_last_negative.c:

```
#include "index_last_negative.h"

int index_last_negative(int s[], int len)
{
   int i;
   for(i = (len - 1); i >= 0; i--)
      if(s[i] < 0)
         break;
   return i;
}
```

multi_between_negative.h:

```
int multi_between_negative(int s[], int len);
```

multi_between_negative.c:

```
#include "index_first_negative.h"
#include "index_last_negative.h"
#include "multi_between_negative.h"

int multi_between_negative(int s[], int len)
{
   int i, k;
   int pr = 1;
   for(i = index_first_negative(s, len), k = index_last_negative(s, len); i < >
      pr *= s[i];
   return pr;

}
```

multi_before_and_after_negative.h:

```
int multi_before_and_after_negative(int s[], int len);
```

multi_before_and_after_negative.c:

```
#include "index_first_negative.h"
#include "index_last_negative.h"
#include "multi_before_and_after_negative.h"

int multi_before_and_after_negative(int s[], int len)
{
   int i, k, a;
   int pr = 1;
```

```
    i = index_first_negative(s, len);
    k = index_last_negative(s, len);
    for(a = 0; a < i; a++)
        pr *= s[a];
    for(a = k; a < len; a++)
        pr *= s[a];
    return pr;
}
menu.c:
#include <stdio.h>
#include <stdlib.h>

#include "index_first_negative.h"
#include "index_last_negative.h"
#include "multi_between_negative.h"
#include "multi_befor_and_after_negative.h"

#define MAX 20

int main()
{
    int arr[MAX];
    int i = 0;
    int n;
    scanf("%d", &n);
    while (getchar() != '\n' && i < MAX){
        scanf("%d", &arr[i]);
        i++;
    }
    switch(n)
```

```
    {
        case 0: printf("%d\n", index_first_negative(arr, i));
                break;
        case 1: printf("%d\n", index_last_negative(arr, i));
                break;
        case 2: printf("%d\n" ,multi_between_negative(arr, i));
                break;
        case 3: printf("%d\n", multi_before_and_after_negative(arr, i));
                break;
        default: puts("Данные некорректны");
                 break;
    }
    return 0;
}
```

Makefile:

```
all: menu.o index_first_negative.o index_last_negative.o multi_between_negative>
        gcc menu.o index_first_negative.o index_last_negative.o multi_between_n>

menu.o: menu.c index_first_negative.h index_last_negative.h multi_between_negat>
        gcc -c menu.c

index_first_negative.o: index_first_negative.c index_first_negative.h
        gcc -c index_first_negative.c

index_last_negative.o: index_last_negative.c index_last_negative.h
        gcc -c index_last_negative.c

multi_between_negative.o: multi_between_negative.c multi_between_negative.h
        gcc -c multi_between_negative.c
```

```
multi_before_and_after_negative.o: multi_before_and_after_negative.c multi_befo>
	gcc -c multi_before_and_after_negative.c
```