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. 0382 _____ Злобин А. С.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Изучить основные принципы сборки программ на языке Си с помощью утилиты make.

Задание.

В текущей директории создайте 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.

Реализуйте 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 отрицательного элемента. (index_first_negative.c)

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

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

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

иначе необходимо вывести строку "Данные некорректны".

Основные теоретические положения.

В данной лабораторной работе были использованы следующие конструкции языка С:

- Функции библиотеки stdio.h:
  - printf()–функция выводит на консоль значине аргумета
  - scanf()–функция ввода данных из консоли
- Функция библиотеки stdlib.h:
  - abs()–функция получения модуля числа
- Циклы:
  - while(){}–каждая итерация проверяет, выполняется ли условие в круглых скобках, если оно верно, то выполняется код в фигурных скобках, а если неверно, то происходит выход из цикла
  - for(){<переменная>; <условие>; <выражение_1>}–код в теле цикла будет исполнятся до тех пор, пока объявленная в цикле переменная будет удовлетворять условию цикла, выражение_1 каким-либо способом меняет значение этой переменной
- Операторы:
  - if(){} ... else{}–если выполняется условия, указанное в круглых скобках, то выполняется код в фигурных скобках после if, иначе–в фигурных скобках после else(elsene является обязательной частью конструкции)
  - switch(<переменная>){casex:... break; ... default:...break;}–от значения переменной в круглых скобках зависит, какой кейс будет выполняться (например, если переменная имеет значение x–выполнится casex). Если же не будет кейса с таким значением, то выполнится код из блока default
- Функции:
  - <тип_функции> имя_функции(<аргумент_1>, ... , <argument_n>) {}–при вызове данной функции в главной(main) функциивыполняется

код в фигурных скобках, а затем возвращает значение оператором return(если тип функции не void)

Также был использован make-файл, который состоит из:

- ○ - списка целей;
- ○ - зависимостей этих целей;
- ○ - команд, которые требуется выполнить, чтобы достичь эту цель.

Содержимое должно выглядеть следующим образом:

цель: зависимости

[tab] команда 3

Первая цель в файле является целью по умолчанию. Для сборки проекта обычно используется цель all, которая находится самой первой

Выполнение работы.

В начале работы создадим файл menu.c, в котором реализуем ввод данных и выбор необходимого действия.

В начале файла необходимо подключить следующие библиотеки:

- stdio.h–используется для подключения ввода-вывода (printf(), scanf())
- stdlib.h–используются для доступа к функции abs(), которая позволяет получить модуль числа

А так же подключить заголовочные файлы:

- index_first_negative.h — содержит объявление функции index_first_negative
- index_last_negative.h — содержит объявление функции index_last_negative
- sum_between_negative.h — содержит объявление функции sum_between_negative
- sum_before_and_after_negative.h — содержит объявление функции sum_before_and_after_negative

Затем объявим переменные:

массив mass[100] — будет хранить входной массив целых чисел

c — переменная типа char, которая будет хранить символ, разделяющий элементы массива

Для того чтобы приступить к решению задачи, необходимо считать данные. Это осуществляется с помощью функции scanf(). В начале считывается переменная kode типа int, определяющая, какую из подзадач необходимо решить. Далее в цикле while осуществляется считывание по две переменные за цикл: mass[i] и c. Цикл while будет выполняться до тех пор, пока переменная c не равна символу новой строки ('\n'). При этом переменная i, отвечающая за обращение к элементам массива, после выхода из цикла будет иметь значение количества элементов массива.

Далее в зависимости от значения kode будет вызываться одна из функций index_first_negative(), index_last_negative(), sum_between_negative(), sum_before_and_after_negative() с помощью оператора switch.

1. Первую подзадачу решает функция index_first_negative(), которая содержится в файле index_first_negative.c. Она получает на вход массив fmass и число max типа int (количество элементов массива, поступившего на вход программе) и возвращает значение типа int. В цикле while значение переменной i (тип int) увеличивается на единицу с каждой новой итерацией. Цикл прерывается когда значение mass[k] становится меньше 0. Функция возвращает номер первого отрицательного элемента.

2. Вторую подзадачу решает функция index_last_negative(), которая содерржится в файле index_last_negative.c. Она получает на вход массив fmass и число max типа int (количество элементов массива, поступившего на вход программе) и возвращает значение типа int. В цикле while значение переменной i (тип int) уменьшается на единицу ,начиная с номера последнего элемента массива (max), с каждой новой итерацией. Цикл прерывается когда

значение mass[k] становится меньше 0. Функция возвращает номер последнего отрицательного элемента.

3. Третью подзадачу решает функция sum_between_negative(), которая содерржится в файле sum_between_negative.c. Она получает на вход массив fmass и число max типа int (количество элементов массива, поступившего на вход программе) и возвращает значение типа int. В функции объявляются переменные типа int:

- k — счётчик для перебора элементов массива
- start — первый элемент суммы
- stop — последний элемент суммы
- summ=0 — значение суммы

  Переменыые start и stop принимают значения функций index_first_negative(fmass, max) и index_last_negative(fmass, max) соответственно. В циикле for складываются элементы от start включительно до stop не включительно. Функция возвращает эту сумму.

4. Четвёртую подзадачу решает функция, которая содерржится в файле sum_before_and_after_negative(). Она получает на вход массив fmass и число max типа int (количество элементов массива, поступившего на вход программе) и возвращает значение типа int. В функции объявляются переменные типа int:

- k — счётчик для перебора элементов массива
- start — первый элемент суммы последних чисел
- stop — последний элемент суммы первых чисел
- summ=0 — значение суммы

Переменые start и stop принимают значения функций i index_last_negative(fmass, max) и ndex_first_negative(fmass, max) соответственно. В циикле for складываются элементы от 0 элемента массива

до stop не включительно и от start включительно до последнего элемента массива (max). Функция возвращает эту сумму.

Далее необходимо создать Makefile, который выполняет сборку. Для удобства работы с проектом будем использовать зависимости.

- Инструкция all имеет зависимость menu

Инструкция menu имеет следующие зависимости:

menu.o index_first_negative.o index_last_negative.o sum_between_negative.o sum_before_and_after_negative.o

и выполняет команду:

gcc menu.o index_first_negative.o index_last_negative.o sum_between_negative.o sum_before_and_after_negative.o -o menu

Выполнение данной инструкции приводит к сборке проекта из необходимых объектных файлов.

С помощью ключа «-o» сообщается название получаемого после выполнения исполняемого файла — menu.

- Инструкция menu.o.

Имеет следующие зависимости: menu.c, max.h, min.h, diff.h, sum.h.

Команда: gcc -c menu.c -o menu.o

Выполнение данной инструкции приводит к созданию объектного файла menu.o.

Выполнение последующих целей так же приводит к созданию объектных файлов для соответсвующих файлов.

- Инструкция clean.

Используется для удаления всех объектных файлов из текущей директории.

Команда: rm *.o.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 2 3 4 -5 2 3 4 -5 4 3 2 -7 -8 5 8 9	4	Программа работает верно
2.	1 1 2 3 4 -5 2 3 4 -5 4 3 2 -7 -8 5 8 9	13	Программа работает верно
3.	2 1 2 3 4 -5 2 3 4 -5 4 3 2 -7 -8 5 8 9	35	Программа работает верно
4.	3 1 2 3 4 -5 2 3 4 -5 4 3 2 -7 -8 5 8 9	40	Программа работает верно
5.	10 1 2 3 4 -5 2 3 4 -5 4 3 2 -7 -8 5 8 9	Данные некорректны	Программа работает верно

Выводы.

В ходе работы был изучен процесс сборки программ на языке Си при помощи утилиты Make.

Была написана программа, считывающая данные с помощью функции scanf(), и выводящей результат с помощью printf(). Далее программа вызывает одну из четырёх функций, в зависимости от значения переменной kode. Это происходило с помощью оператора switch. Если значение kode = 0, то вызывалась функция index_first_negative. Если значение kode = 1, то вызывалась функция index_last_negative. Если значение kode = 2, то вызывалась функция sum_between_negative. Если значение kode = 3, то вызывалась функция sum_before_and_after_negative. После этого выполнялась соответствующая подзадача.

Каждая функция хранится в отдельном файле. Для каждой функции создан файл с расширением *.c, в котором хранится определение функции и заголовочный файл, в котором находится объявление функции. Основная функция main находится в файле menu.c Кроме того, был разработан make-

файл, в котором расположены инструкции по сборке программы, указаны зависимости этих инструкций и команды, которые необходимо выполнить. Результатом работы утилиты Make является исполняемый файл menu.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: menu.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_negative.h"
#include "index_last_negative.h"
#include "sum_between_negative.h"
#include "sum_before_and_after_negative.h"
int main()
{
	int mass[100] = { 0 };
	int kode, i=0, result;
	scanf ("%d", &kode);
	char c = ' ';
	while (c != '\n')
	{
		scanf("%d%c", &mass[i], &c);
		i++;
	}
	switch (kode)
	{
		case 0:
			result = index_first_negative(mass, i);
			break;
		case 1:
			result = index_last_negative(mass, i);
			break;
		case 2:
			result = sum_between_negative(mass, i);
			break;
		case 3:
			result = sum_before_and_after_negative(mass, i);
			break;
		default:
		printf ("%s", "Данные некорректны");
		break;
	}
	if (kode >=0 && kode<=3)
		printf ("%d", result);
	return 0;
}
```

Название файла: index_first_negative.c

```
#include <stdio.h>
int index_first_negative(int fmass[], int max)
{
	int i, k=-1;
	for (i=0; i<max; i++)
		if (fmass[i] < 0)
			{
```

```
                    k=i;
                    break;

                }
    return k;
}
```

Название файла: index_first_negative.h

```
int index_first_negative(int fmass[], int max);
```

Название файла: index_last_negative.c

```
#include <stdio.h>
int index_last_negative(int fmass[], int max)
{
    int k = -1, i;
    for (i=max-1; i>=0; i--)
        if (fmass[i]<0)
        {
            k=i;
            break;
        }
    return k;
}
```

Название файла: index_last_negative.h

```
int index_last_negative(int fmass[], int max);
```

Название файла: sum_between_negative.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_negative.h"
#include "index_last_negative.h"
int sum_between_negative(int fmass[], int max)
{
    int k, start, stop, summ=0;
    start = index_first_negative(fmass, max);
    stop = index_last_negative(fmass, max);
    for (k = start; k<stop; k++)
        summ += abs(fmass[k]);
    return summ;
}
```

Название файла: sum_between_negative.h

```
int sum_between_negative(int fmass[], int max);
```

Название файла: sum_before_and_after_negative.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_negative.h"
#include "index_last_negative.h"
int sum_before_and_after_negative(int fmass[], int max)
```

```
{
      int k, start, stop, summ=0;
      stop = index_first_negative(fmass, max);
      start = index_last_negative(fmass, max);
      for (k = 0; k<stop; k++)
            summ += abs(fmass[k]);
      for (k=start; k<=max; k++)
            summ += abs(fmass[k]);
      return summ;
}
```

Название файла: sum_before_and_after_negative.h

```
int sum_before_and_after_negative(int fmass[], int max);
```

Название файла: Makefile:

```
all : menu

menu:    menu.o    index_first_negative.o    index_last_negative.o
sum_between_negative.o sum_before_and_after_negative.o
      gcc   menu.o   index_first_negative.o   index_last_negative.o
sum_between_negative.o sum_before_and_after_negative.o -o menu
menu.o:   menu.c   index_first_negative.h   index_last_negative.h
sum_between_negative.h sum_before_and_after_negative.h
      gcc -c menu.c -o menu.o
index_first_negative.o: index_first_negative.c
      gcc -c index_first_negative.c -o index_first_negative.o
index_last_negative.o: index_last_negative.c
      gcc -c index_last_negative.c -o index_last_negative.o
sum_between_negative.o:                    sum_between_negative.c
index_first_negative.h index_last_negative.h
      gcc -c sum_between_negative.c -o sum_between_negative.o
sum_before_and_after_negative.o:  sum_before_and_after_negative.c
index_first_negative.h index_last_negative.h
      gcc       -c       sum_before_and_after_negative.c       -o
sum_before_and_after_negative.o
clean:
      rm *.o
```