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

Студентка гр. 0382 _____ Михайлова О.Д.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Изучение сборки программ в СИ с помощью make-файла.

Задание.

Вариант 6.

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.

Препроцессор – это программа, которая подготавливает код программы для передачи ее компилятору.

Команды препроцессора называются директивами и имеют следующий формат: #ключевое_слово параметры.

Основные действия, выполняемые препроцессором:

- Удаление комментариев
- Включение содержимого файлов (*#include*)
- Макроподстановка (*#define*)
- Условная компиляция (*#if, #ifdef, #elif, #else, #endif*)

Компиляция - процесс преобразования программы с исходного языка высокого уровня в эквивалентную программу на языке более низкого уровня (в частности, машинном языке).

Компилятор - программа, которая осуществляет компиляцию.

Линковщик (компоновщик) принимает на вход один или несколько объектных файлов и собирает по ним исполняемый модуль. Работа компоновщика заключается в том, чтобы в каждом модуле определить и связать ссылки на неопределённые имена.

Сборка проекта – это процесс получения исполняемого файла из исходного кода.

Сборка проекта вручную может стать довольно утомительным занятием, особенно, если исходных файлов больше одного и требуется задавать некоторые параметры компиляции/линковки. Для этого используются Makefile - список инструкций для утилиты make, которая позволяет собирать проект сразу целиком.

Любой make-файл состоит из

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

цель: зависимости

[tab] команда

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

Выполнение работы.

Переменные:

#define N 100 – макрос, обозначающий максимальный размер массива;

Int arr[N] – массив;

Int arr_size – переменная, в которую записывается размер массива;

Char sym – символ-разделитель (пробел);

Int command – переменная для считывания значения;

Int res0 – переменная, в которую записывается результат при вводе значения 0;

Int res1 – переменная, в которую записывается результат при вводе значения 1;

Int res2 – переменная, в которую записывается результат при вводе значения 2;

Int res3 – переменная, в которую записывается результат при вводе значения 3;

Int i – счетчик в цикле;

Int sum – переменная, в которую записывается сумма элементов массива;

Int first – переменная, в которую записывается индекс первого отрицательного элемента массива;

Int last – переменная, в которую записывается индекс последнего отрицательного элемента массива.

Функции:

Функция int index_first_negative (int arr[N], int arr_size) принимает на вход массив arr размера arr_size, находит и возвращает индекс первого отрицательного элемента массива.

Объявление функции index_first_negative содержится в файле index_first_negative.h.

Определение функции index_first_negative содержится в файле index_first_negative.c, который включает стандартную библиотеку stdio.h и содержимое файла index_first_negative.h с помощью директивы #include.

Функция int index_last_negative (int arr[N], int arr_size) принимает на вход массив arr размера arr_size, находит и возвращает индекс последнего отрицательного элемента массива.

Объявление функции index_last_negative содержится в файле index_last_negative.h.

Определение функции index_last_negative содержится в файле index_last_negative.c, который включает стандартную библиотеку stdio.h и содержимое файла index_last_negative.h с помощью директивы #include.

Функция int sum_between_negative (int arr[N], int arr_size) принимает на вход массив arr размера arr_size. Переменным first и last присваиваются

значения индексов первого и последнего отрицательных элементов массива соответственно. Далее функция считает и выводит сумму модулей элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент).

Объявление функции sum_between_negative содержится в файле sum_between_negative.h.

Определение функции sum_between_negative содержится в файле sum_between_negative.c, который включает стандартные библиотеки stdio.h и stdlib.h и содержимое файлов index_first_negative.h, index_last_negative.h, sum_between_negative.h с помощью директивы #include.

Функция int sum_before_and_after_negative (int arr[N], int arr_size) принимает на вход массив arr размера arr_size. Переменным first и last присваиваются значения индексов первого и последнего отрицательных элементов массива соответственно. Далее функция сначала считает сумму модулей элементов массива, расположенных до первого отрицательного элемента (не включая элемент), затем прибавляет сумму модулей элементов массива, расположенных после последнего отрицательного элемента (включая элемент) и выводит общую сумму.

Объявление функции sum_before_and_after_negative содержится в файле sum_before_and_after_negative.h.

Определение функции sum_before_and_after_negative содержится в файле sum_before_and_after_negative.c, который включает стандартные библиотеки stdio.h и stdlib.h и содержимое файлов index_first_negative.h, index_last_negative.h, sum_before_and_after_negative.h с помощью директивы #include.

В функции main на вход поступает сначала значение, которое обозначает команду, выбранную пользователем, затем массив. Оператор множественного выбора switch вызывает нужную команду и выводит результат. Если на вход поступает значение, не указанное в операторе switch,

то выводится сообщение «Данные некорректны». При корректной работе программа возвращает 0.

Функция main находится в файле menu.c, который включает стандартные библиотеки stdio.h и stdlib.h и все вышеописанные файлы с расширением .h с помощью директивы #include.

Утилита make собирает программу с помощью Makefile, в котором описаны инструкции для компиляции программы.

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 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	3	Получили индекс первого отрицательного элемента массива.
2.	2 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	226	Получили сумму модулей элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент).
3.	1 4 -5 76 34 2 6 1 -8 -2 3 43 17 -13 4	12	Получили индекс последнего отрицательного элемента массива.
4.	4 4 -5 76 34 2 6 1 -8 -2 3 43 17 -13 4	Данные некорректны	В операторе switch не указан случай, когда значение равно 4, поэтому получили «Данные некорректны».
5.	3 5 -6 7 24 35 9 -8 4 -1 -12 4 -5 6	16	Получили сумму модулей элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и

			после последнего отрицательного (включая элемент).

Выводы.

Был изучен способ сборки программы в СИ с помощью make-файла.

Разработана программа, выполняющая считывание с клавиатуры исходных данных и команды пользователя. Для выбора команды используется оператор switch. Для обработки команд в функциях используются условные операторы if и while, а также цикл for. Разработан Makefile, с помощью которого утилита make собирает программу из нескольких файлов.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: index_first_negative.h

```
#define N 100

int index_first_negative (int arr[N], int arr_size);
```

Название файла: index_first_negative.c

```
#include <stdio.h>
#include "index_first_negative.h"
#define N 100

int index_first_negative (int arr[N], int arr_size){
    for (int i=0; i<arr_size; i++){
        if (arr[i]<0){
            return i;
        }
    }
}
```

Название файла: index_last_negative.h

```
#define N 100

int index_last_negative (int arr[N], int arr_size);
```

Название файла: index_last_negative.c

```
#include <stdio.h>
#include "index_last_negative.h"
#define N 100

int index_last_negative (int arr[N], int arr_size){
    for (int i=(arr_size-1); i>=0; i--){
        if (arr[i]<0){
            return i;
        }
    }
}
```

Название файла: sum_between_negative.h

```
#define N 100

int sum_between_negative (int arr[N], int arr_size);
```

Название файла: sum_between_negative.c

```
#include <stdio.h>
#include <stdlib.h>
#include "sum_between_negative.h"
#include "index_first_negative.h"
#include "index_last_negative.h"
#define N 100

int sum_between_negative (int arr[N], int arr_size){
```

```
    int sum = 0;
    int first, last, i;
    first = index_first_negative (arr, arr_size);
    last = index_last_negative (arr, arr_size);

    for (i=first; i<last; i++){
        sum=sum+abs(arr[i]);
    }

    return sum;
}
```

Название файла: sum_before_and_after_negative.h

```
#define N 100

int sum_before_and_after_negative (int arr[N], int arr_size);
```

Название файла: sum_before_and_after_negative.c

```
#include <stdio.h>
#include <stdlib.h>
#include "sum_before_and_after_negative.h"
#include "index_first_negative.h"
#include "index_last_negative.h"
#define N 100

int sum_before_and_after_negative (int arr[N], int arr_size){
    int sum = 0;
    int first, last, i;
    first = index_first_negative (arr, arr_size);
    last = index_last_negative (arr, arr_size);

    for (i=0; i<first; i++){
        sum=sum+abs(arr[i]);
    }

    for (i=last; i<arr_size; i++){
        sum=sum+abs(arr[i]);
    }

    return sum;

}
```

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"
#define N 100

int main(){
    int arr[N];
    int arr_size = 0;
    char sym = ' ';
    int res0, res1, res2, res3;
```

```
        int x;
        scanf ("%d", &x);

        while ((arr_size<N) && (sym == ' ')){
            scanf ("%d%c", &arr[arr_size++], &sym);
        }

        switch (x){
            case 0:
            res0=index_first_negative (arr, arr_size);
            printf ("%d\n", res0);
            break;
            case 1:
            res1=index_last_negative (arr, arr_size);
            printf ("%d\n", res1);
            break;
            case 2:
            res2=sum_between_negative (arr, arr_size);
         printf ("%d\n", res2);
            break;
            case 3:
            res3=sum_before_and_after_negative (arr, arr_size);
            printf ("%d\n", res3);
         break;
            default:
            printf ("Данные некорректны\n");
        }
        return 0;
    }
```

Название файла: Makefile

```
    all:    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
    index_first_negative.o: index_first_negative.c
            gcc -c index_first_negative.c -std=c99
    index_last_negative.o: index_last_negative.c
            gcc -c index_last_negative.c -std=c99
    sum_between_negative.o:                    sum_between_negative.c
index_first_negative.h index_last_negative.h
            gcc -c sum_between_negative.c -std=c99
    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 -std=c99
    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 -std=c99
```