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 проектов в языке СИ

Студентка гр. 2384 _____ Валеева А. А.

Преподаватель _____ Гаврилов А.В.

Санкт-Петербург
2022

Цель работы.

Изучение процесса сборки программ, написанных на языке Си на примере использования make-файлов.

Задание.

Вариант 5:

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 : максимальное по модулю число в массиве. (abs_max.c)

1 : минимальное по модулю число в массиве. (abs_min.c)

2 : разницу между максимальным по модулю и минимальным по модулю элементом. (diff.c)

3 : сумму элементов массива, расположенных после максимального по модулю элемента (включая этот элемент). (sum.c), иначе необходимо вывести строку "Данные некорректны".

Ошибкой в данном задании считается дублирование кода! Подсказка: функция нахождения модуля числа находится в заголовочном файле stdlib.h стандартной библиотеки языка Си. При выводе результата, не забудьте символ переноса строки

Основные теоретические положения.

Для выполнения лабораторной работы требуется изучение Make-файлов. Make-файл - это текстовый файл, в котором определенным образом описаны правила сборки приложения. Для работы с make-файлами используется утилита make, которая позволяет не компилировать файлы дважды, если это не требуется.

Выполнение работы.

Создаются файлы .c для определения каждой функции, а также к каждой функции создаётся заголовочный файл .h для её объявления. В каждый файл .h добавляется директива *#pragma once* для предотвращения повторного включения заголовочных файлов, определение соответствующей функции и необходимые

зависимости. В файлы .c записываются определения функций и подключаются соответствующие
заголовочные файлы. В файл *menu.c* подключаются все заголовочные файлы: *abs_max.h, abs_min.h, diff.h, sum.h*, а также *stdio.h.*

Используя оператор *switch*, вызывается необходимая функция в зависимости от значения вводимой переменной *value* типа *int*. Если значение переменной будет некорректно, выполнение программы завершится, и пользователь получит сообщение «Данные некорректны».

Рассмотрим функции:

int abs_min(int array[], int size) – На вход принимает массив типа *int* и переменная *size* типа *int*, равная размеру массива. Функция возвращает минимальный по модулю элемент массива.

int abs_max(int array[], int size) – На вход принимает массив типа int и переменная *size* типа *int*, равная размеру массива. Функция возвращает максимальный по модулю элемент массива.

int diff(int array[], int size) – На вход принимает массив типа int и переменная *size* типа *int*, равная размеру массива. Функция возвращает разницу между максимальным и минимальным по модулю элементами. Так как используем вызов функции *abs_min* и *abs_max*, то подключаем заголовочные файлы соответствующих функций: *abs_min.h* и *abs_max.h*.

int sum(int array[], int size) – На вход принимает массив типа int и переменная *size* типа *int*, равная размеру массива. Функция возвращает сумму элементов, стоящих после максимального по модулю элемента, включая его. Используем вызов функции *abs_max*, для ее использования подключаем заголовочный файл *abs_max.h.*

Для компиляции был написан Makefile.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования.

value	Входные данные	Выходные данные	Комментарии
1.	0 -28 26 30 22-13-28 3 -30 12 8 10 -19 -26 11 -6 -18 -3 -2 -26 18 8 -19 -17 -11 -12 -23 19 -16 -11 9	30	Верный ответ
2.	1 4 88 99 77 6 4 -50 -100	4	Верный ответ
3.	2 89 -67 89 3 8 6 -87 67 5	86	Верный ответ
4.	3 98 66 44 5 34 8 94 3	352	Верный ответ
5.	5 88 6 5 4 3 98 76 4	Данные некорректны	Нет команды под номером 5.

Вывод.

Изучен процесс сборки программ в языке Си. Также был написан Makefile, позволяющий собрать программу, состоящую из нескольких файлов.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Makefile

```
all: menu.o abs_max.o abs_min.o diff.o sum.o
	gcc menu.o abs_max.o abs_min.o diff.o sum.o -o menu
menu.o: menu.c abs_max.h abs_min.h diff.h sum.h
	gcc -c -std=c99 menu.c
abs_max.o: abs_max.c abs_max.h
	gcc -c -std=c99 abs_max.c
abs_min.o: abs_min.c abs_min.h
	gcc -c -std=c99 abs_min.c
diff.o: diff.c diff.h
	gcc -c -std=c99 diff.c
sum.o: sum.c sum.h
	gcc -c -std=c99 sum.c
```

Название файла: menu.c

```
#include <stdio.h>
#include <stdlib.h>
#include "sum.h"
#include "diff.h"
#include "abs_min.h"
#include "abs_max.h"
#define MAX_SIZE(100)

int main(){
	int value;
	int arr[LIMIT_SIZE];
	int current_size = 0;
	char symbol;
	scanf("%d%c", &value, &symbol);
	do{
		scanf("%d%c", &arr[current_size ++], &symbol);
		}while(symbol != '\n' && current_size < LIMIT_SIZE);
	switch (value){
		case 0:
		{
			printf("%d\n", abs_max(arr, current_size));
			break;
		}
		case 1:
		{
			printf("%d\n", abs_min(arr, current_size ));
			break;
		}
		case 2:
		{
			printf("%d\n", diff(arr, current_size));
			break;
```

```
            }
            case 3:
            {
                  printf("%d\n", sum(arr, current_size));
                  break;
            }
            default:
            {
                  printf("Данные некорректны");
                  break;
            }
      }
}
```

Название файла: abs_max.c

```
#include <stdio.h>
int abs_max(int array[], int size){
      int max_arr = array[0];
      for (int index = 1; index < size; index ++){
            if (abs(array[index]) > abs(max_arr)){
                  max_arr = array[index];
            }
      }
      return max_arr;
}
```

Название файла: abs_max.h

```
#pragma once
int abs_max(int array[], int size);
```

Название файла: abs_min.c

```
#include <stdio.h>
int abs_min(int array[], int size){
      int min_arr = array[0];
      for (int index = 1; index < size; index ++){
            if (abs(array[index]) < abs(min_arr)){
                  min_arr = array[index];
            }
      }
      return min_arr;
}
```

Название файла: abs_min.h

```
#pragma once
int abs_min(int array[], int size);
```

Название файла: diff.c

```
#include <stdio.h>
#include "abs_min.h"
#include "abs_max.h"
int diff(int array[], int size){
      int max = abs_max(array, size);
      int min = abs_min(array, size);
      return max-min;
}
```

Название файла: diff.h

```
#pragma once
int diff(int array[], int size);
```

Название файла: sum.c

```
#include <stdio.h>
#include “abs_max.h”
int sum(int array[], int size){
      int max = abs_max(array, size);
      int sum = 0;
      int ready = 0;
      for (int i = 0; i < size; i++){
            if (abs(array[i]) == abs(max)){
                  ready = 1;
            }
            if (ready){
                  sum += array[i];
            }
      }
      return sum;
}
```

Название файла: sum.h

```
#pragma once
int sum(int array[], int size);
```