**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. 1304 _____ Макки К.Ю

Преподаватель _____ Чайка К.В.

Санкт-Петербург
2021

**Цель работы.**

Изучение способы использования и работы с makefile на терминале

**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 : максимальное число в массиве. (max.c)

1 : минимальное число в массиве. (min.c)

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

3 : сумму элементов массива, расположенных до минимального элемента. (sum.c)

иначе необходимо вывести строку "Данные некорректны".

Вариант 2.

**Основные теоретические положения.**

В данной лабораторной работе использовалась библиотека stdio.h и сборка программ с помощью makefile.

**Выполнение работы.**

Необходимо написать программу, которая может выполнять 4 разных функции с массивом номеров, но на выбор пользователя. В этом массиве есть

2 части, выбор функции в виде одного числа и сам массив размером не больше 100 чисел. Все числа должны быть разделены пробелом, а символ перевод строки обозначает конец ввода пользователя. Программа начинается с функции arr_imp(array input). Это функция
принимает один 1 аргумент, arr[] где будет храниться введенные данные. Мы объявляем 2 переменные sep(separator) и n(number). В переменной sep вида char будет храниться символ разделения между числами массива, а в n будет храниться количество чисел в массиве в виде int. Потом пишем цикл for который принимает 3 условия. Первое, int i=0 который будет хранить в себе количество раз цикл повторяется, второй i< 100 который говорит программе 2повторять цикл пока i меньше 100, последний i++ который прибавляет значение i каждый раз цикл повторяется. После for у нас функция scanf которая принимает данные от пользователя и ставит их в обозначенные переменные. После приема данных мы прибавляем n чтобы к концу цикла узнать количество раз он повторился и назначит это число как размер массива. Мы используем оператор if с условием (sep == '\n') в случае это условие верное цикл сразу заканчиваться. В конце функция arr_inp возвращает нам число n который будет хранить точный размер массива.
Все остальные функции будут храниться в разных файлах эти файлы будут разделены таким образом. В файле max_num.c будет толбко вторая функция без #include других файлов вотому что другие функции кроме max_num(maximum number) которая принимает 2
аргумента первый это массив arr и размер массива n. В функции мы объявляем переменную max = arr[1], первое число в массиве. Потом используем цикл for который будет проходить через каждый число этого массива и сравнивать его с max если это число больше тогда max будет равен этому числу. В цикле for мы ставим index i = 1 потому что первый число введенных данных не часть массива. Функция возвращает переменную max.

Третья функция будет храниться в файле min_num.c в котором min_num (минимальный number) это функция тоже
принимает 2 параметра такие же как прошлая. Это Функция находит самый маленький номер в массиве. Мы создаём переменную min = arr[1] первый элемент массива. Используя цикл for с такими же условиями как в прошлой функции. Внутри цикла for мы используем if с условием если число массива меньше переменной min этот элемент становиться минимальным элементом.
Четвертая функция diff_num(difference number) будет в файле diff_num.c в этом файле надо будет использовать #include "max_num.h" и #include "min_num.h" в которых будет объявление первой и второй фуккции которые нужны для вывода ответа этой функци .В этой функции начинаем с объявление переменную max в которой вызываем функцию max_num(arr,n). Это переменная будет равна максимальному числу массива. Потом делаем то же самое с переменной min = min_num(arr,n) , но с минимальным номером массива. Третья переменная diff будет равна махального - минимального числа. И в конце функция возвращает эту переменную diff.
Пятая функция sum_num(sum numbers) будет в файле sum_num.c с #include "min_num.h" где будет объявлена фукйция min_num .это фунция должна вывести сумму всех чисел до минимального чесал массива. Функция принимает те же самые 2
параметры как предыдущее функции. Начинаем и создаём переменную sum=0 это переменная будет хранить в себе сумму чисел. Ещё мы создаем переменную min = min_num(arr,n) а этой переменной мы вызываем функцию чтобы потом сравнивать числа массива с минимальным числом и узнать когда заканчивать цикл. О цикле, мы используем цикл for в котором лежит if которая, сравнивает числа с минимальным числом и когда числа равны цикл заканчивается если условие не верное число массива с индексом i добавляется в sum и становиться им. Это функция возвращает нам

переменную sum.

Кроме файлов с .c есть ещё фйлы .h где будет храниться вызовы нужных функций. Почему .h потому что этот тип файла который можно использовать в #include.
Makefile нужен чтобы собирать компилировать и link все нужные файлы вместе. Он в конце выдает .o файлы где машинный код.

Последняя функция main() и самая главная. В этой функции мы обяляем переменную arr[100] массив с максимальным размером 100 элементов вида int. После этого нам нужно объявить переменную n которая равна вызову функции arr_inp это функция возвращает длина этого массива. Но главная функция ее вызова это дать возможность пользователю ввести числа в массив. Потом мы создаём switch который сравнивает элемент массива с индексом 0 с разными case. Если arr[0]==0 функция печатает на экран результат функции max_num и случается break этого switch. Если arr[0]==1 функция печатает на экран результат функции min_num и ломается функция. И таким образом если arr[0]== 2 или 3 печататься на экране diff_num или sum_num соответственно. В случае arr[0] неравен не какому из case тогда выводиться на экране "Данные некорректны".

**Выводы.**

Я изучил процесс сборки программ и как это происходит в язые Си, ещё я использовал основные команды для работы с makefile.

## ПРИЛОЖЕНИЕ А

## ИСХОДНЫЙ КОД ПРОГРАММЫ

**makefile:**

```
all: menu.o max_num.o min_num.o diff_num.o sum_num.o
	gcc menu.o max_num.o min_num.o diff_num.o sum_num.o -o menu
menu.o: menu.c max_num.h min_num.h diff_num.h sum_num.h
	gcc -c menu.c
ax_num.o: max_num.c max_num.h
	gcc -c max_num.c
min_num.o: min_num.c min_num.h
	gcc -c min_num.c
diff_num.o: diff_num.c max_num.h min_num.h diff_num.h
	gcc -c diff_num.c
sum_nums.o: sum_num.c min_num.h sum_num.h
	gcc -c sum_num.c
clean:
	rm -r menu menu.o max_num.o min_num.o diff_num.o sum_num.o
```

**menu.c:**

```
#include<stdio.h>
#include<stdlib.h>
#include "max_num.h"
#include "min_num.h"
#include "diff_num.h"
#include "sum_num.h"


int arr_inp(int arr[]){
	char sep;
	int n = 0;
	for (int i = 0; i < 100; i++){
		scanf("%d%c", &arr[i], &sep);
		n++;
			if(sep == '\n'){
				break;
			}
	}
	return n;
}

int main(){
	int arr[100];
	int n = arr_inp(arr);
	switch(arr[0]) {
		case 0:
		  printf("%d\n", max_num(arr, n));
		  break;
		case 1:
		  printf("%d\n", min_num(arr, n));
		  break;
		case 2:
		  printf("%d\n", diff_num(arr, n));
		  break;
		case 3:
```

```
            printf("%d\n", sum_num(arr, n));
            break;
          default:
            printf("Данные некорректны\n");
      }
      return 0;
}
```

**max_num.c:**

```
int max_num(int arr[], int n){
    int max = arr[1];
    for(int i=1;i < n;i++){
      if(arr[i]>=max){
      max = arr[i];
      }
    }
    return max;
}
```

**max_num.h:**

```
int max_num(int arr[], int n);
```

**min_num.c:**

```
int min_num(int arr[], int n){
    int min = arr[1];
    for(int i=1;i < n;i++){
      if(arr[i]<=min){
      min = arr[i];
      }
    }
    return min;
}
```

**min_num.h:**

```
int min_num(int arr[], int n);
```

**diff_num.h:**

```
#include "max_num.h"
#include "min_num.h"

int diff_num(int arr[], int n){
    int max = max_num(arr, n);
    int min = min_num(arr, n);
    int diff = max - min;
    return diff;
}
```

**diff_num.h:**

```
int diff_num(int arr[], int n);
int max_num(int arr[], int n);
int min_num(int arr[], int n);
```

**sum_num.c:**

```
#include "min_num.h"

int sum_num(int arr[], int n){
    int sum = 0;
    int min = min_num(arr , n);
    for(int i = 1; i < n; i++){
```

```
            if(arr[i] == min){
                  break;
            }
            sum = sum + arr[i];

      }
      return sum;
}
```

**sum_num.h:**

```
int sum_num(int arr[], int n);
int min_num(int arr[], int n);
```

## ПРИЛОЖЕНИЕ Б

## ТЕСТИРОВАНИЕ

Таблица Б.1 - Примеры тестовых случаев

| № п/п | Входные данные | Выходные данные | Комментарии |
|---|---|---|---|
| 1. | 2 9 2 4 7 6 | 7 | Верный ответ |
| 2. | 0 9 2 4 7 6 | 9 | Верный ответ |
| 3. | 3 9 2 4 7 6 | 9 | Верный ответ |
| 4. | 1 9 2 4 7 6 | 2 | Верный ответ |