МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ

ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине

«Программирование» Тема:

Сборка программ в Си

Студент гр. 2384 ___ Кузьминых Е.М.

Преподаватель ___ Гаврилов А.В.

Санкт-Петербург

2022

Цель работы.

Написать программу на языке программирования C, выполняющую различную обработку данных в зависимости от введенных команд, разбив код по функциям на файлы, а также используя makefile.

Задачи.

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 нулевого элемента. (index_first_zero.c)

1 : индекс последнего нулевого элемента. (index_last_zero.c)

2 : Найти сумму модулей элементов массива, расположенных от первого нулевого элемента и до последнего. (sum_between.c)

3 : Найти сумму модулей элементов массива, расположенных до первого нулевого элемента и после последнего. (sum_before_and_after.c)

иначе необходимо вывести строку "Данные некорректны".

Выполнение работы.

В начале создадим главный файл menu.c, в нем мы подключим библиотеки stdio.h и stdlib.h с помощью #include, а также с помощью #define

объявим константу Array_length для размера массива. После создадим файлы расширений .c и .h, которые будут содержать функции, необходимые для работы программы:

Index_first_zero – принимает количество элементов в массиве, а также сам массив с числами. Далее, с помощью цикла for функция “пробегает” каждый элемент массива. Если элемент равен нулю, функция выводит его индекс и заканчивает работу. В случае некорректных данных функция возвращает -100.

Index_last_zero – работает аналогично функции index_last_zero, принимает массив и количество элементов в нем, после “пробегает” каждый элемент с помощью цикла for и возвращает индекс последнего нуля в массиве. В случае некорректных данных функция возвращает -100.

Sum_between – эта функция находит сумму модулей элементов, расположенных между первым и последним нулем в массиве. Для этого задействуются уже созданные отдельно функции index_first_zero и index_last_zero. Для их использования они подключаются к файлу к файлу sum_between.c с помощью #include (именно index_first_zero.h и index_last_zero.h).
Далее с помощью if проверяется, корректны ли значения этих функций (если ли нули в введенном массиве). Если все верно, то создаем переменную sum, а с помощью цикла for мы проходим по массиву начиная с первого, заканчивая последним нулем и прибавляем к переменной sum модули элементов, расположенных между нулями. Возвращает сумму модулей элементов между нулями, в случае некорректных данных возвращает -100.

Sum_before_and_after – функция с функционалом, “противоположным” sum_between. Также принимает массив введенных

элементов и количество элементов в массиве, реализует index_first_zero и index_last_zero (неободимо “подключение” к файлу sum_before_and_after.c файлов index_first_zero.h и index_last_zero.h с помощью #include) и проверяет их на корректность данных. Но при использовании цикла for она считает сумму модулей элементов, расположенных до первого нуля и после последнего нуля. Для этого используется переменная sum и два цикла for, первый суммирует все значения от первого элемента массива до первого нуля, второй суммирует все значения от последнего нуля до последнего элемента в массиве. Возвращает переменную sum, в случае некорректных данных возвращает -100.

После создания файлов с функциями, мы “подключаем” их заголовочные файлы с помощью #include к нашему главному файлу menu.c. В нем находится функция main, в которой объявлен массив array, куда будут считываться числа, переменные count_elements, last_symbol и operation. Они определяют количество элементов в массиве, последний символ, введенный пользователем, а также номер операции, который должна выполнить программа. Далее с помощью scanf мы считываем число и сохраняем его в переменную operation. После, с помощью цикла do while мы считываем массив элементов до того момента, пока пользователь не введет enter.

Затем, с помощью оператора switch case мы выводим на экран различную функцию, зависящую от числа, которое ввел пользователь. В случае неверно введенных данных оператор выведет “Данные некорректны”.

После того, как мы написали все функции и подключили их между собой, создадим makefile, в котором укажем зависимости между файлами для компиляции программы.

Тестирование.

№	Входные данные	Выходные данные	Комментарий
1	0 43 54 85 66 0 -23 43 4 0	5	Ответ верный
2	1 45 -34 23 6034 0 37 0 96	6	Ответ верный
3	2 324 43 45 0 1 2 3 0 343 954	6	Ответ верный
4	3 1 2 3 0 23 45 -334 0 4 -5 6	21	Ответ верный

Вывод.

В ходе выполнения лабораторной работы были изучены процессы сборки программ в языке C, разбиение кода на отдельные файлы, а также использован makefile для успешной компиляции программы.

Приложение 1. Исходный код программы.

Index_first_zero.h

```
#include <stdio.h>
int index_first_zero( int count_elements, int array[]);
```

Index_first_zero.c

```
#include <stdio.h>
#include "index_first_zero.h"
int index_first_zero( int count_elements, int array[]) //Функция, возвращающая индекс первого элемента, равного нулю.
{
    int first_index = 0;
    for (int i = 0; i < count_elements; i++)
    {
        if (array[i]==0)
        {

            first_index = i;
            return first_index;

        }
    }
    return -100;
}
```

Index_last_zero.h

```
#include <stdio.h>
int index_last_zero(int count_elements, int array[]);
```

Index_last_zero.c

```
#include <stdio.h>
#include "index_last_zero.h"

int index_last_zero(int count_elements, int array[]){ //Функция, возвращающая индекс последнего элемента, равного нулю.
    int last_index = 0;
    for (int t = 0; t < count_elements; t++)
```

```
    {
        if (array[t]==0)
        {
            last_index = t;


        }


    }

    return last_index;
}
```

Sum_between.h

```
#include <stdio.h>
#include <stdlib.h>
int sum_between(int count_elements, int array[]);
```

Sum_between.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_zero.h"
#include "index_last_zero.h"
int sum_between(int count_elements, int array[]){//Фунция, возвращающая
сумму модулей чисел, расположенных между первым и последним нулем.
    int zero_first_index = index_first_zero(count_elements,array);
    int    zero_last_index=    index_last_zero(count_elements,array);//
Используем предыдущие функции для нахождения индексов нулей.
    int sum = 0;
    if       (zero_first_index!=-1       &&       zero_last_index!=-1&&
zero_first_index!=zero_last_index)//Проверка корректных данных.
    {
        for (int i =zero_first_index; i < zero_last_index; i++)
        {
            sum+=abs(array[i]);

        }
```

```
        return sum;
    }     if (zero_first_index==zero_last_index) // Возвращаем сумму
модулей при верных данных, в противном случае возвращаем 0  -> если 0 в
массиве 1,
    {                                          // -1 -> при некорректных
данных.
        return 0;
    }   else{
        return -100;
    }

}
```

Sum_before_and_after.h

```
#include <stdio.h>
#include <stdlib.h>
int sum_before_and_after(int count_elements, int array[]);
```

Sum_before_and_after.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_zero.h"
#include "index_last_zero.h"

int sum_before_and_after(int count_elements, int array[]){//Функция,
возвращающая сумму модулей чисел, расположенных до первого нуля и после
последнего.
    int zero_first_index = index_first_zero(count_elements,array);
    int zero_last_index= index_last_zero(count_elements,array);

    int sum=0;
    if   (zero_first_index!=-1   &&   zero_last_index!=-1)//Аналогично
sum_between используем функции для нахождения индекса нулей, проверяем
корректность данных
    {                                          //И с помощью двух
циклов for считываем сумму модулей.
        for (int i = 0; i < zero_first_index; i++)
        {
```

```
            sum+=abs(array[i]);
        }
        for (int i =zero_last_index; i < count_elements; i++)
        {
            sum+=abs(array[i]);
        }
        return sum;
    }else{
        return -100;
    }

}
```

Menu.c

```
#include <stdio.h>
#include <stdlib.h>
#include "index_first_zero.h"
#include "index_last_zero.h"
#include "sum_between.h"
#include "sum_before_and_after.h"
#define Array_length 100
int main(){
    int array[Array_length]; //Обозначим массив, куда будем сохранять числа, переменные, отвечающие за кол-во элементов и последний элемент.
    int count_elements =0;
    char last_symbol;
    int operation;
    scanf("%d", &operation); //Принимаем от пользователя число, отвечающее за выполняемую операцию.

    do //Считываем массив чисел
    {
        scanf("%d%c", &array[count_elements], &last_symbol);
        count_elements++;
    } while (last_symbol!='\n' && count_elements<Array_length);

    switch (operation) //Вызываем различные функции в зависимости от числа, которое ввел пользователь.
    {
```

```
        case 0:
            printf("%d\n",index_first_zero(count_elements,array));
            break;

        case 1:
            printf("%d\n",index_last_zero(count_elements,array));
            break;
        case 2:
            printf("%d\n",sum_between(count_elements,array));
            break;

        case 3:
            printf("%d\n",sum_before_and_after(count_elements,array));
            break;
        default:
            printf("Данные некорректны");
    }

    return 0;
}
```

Makefile

```
all: menu.o index_first_zero.o index_last_zero.o sum_between.o
sum_before_and_after.o
     gcc menu.o index_first_zero.o index_last_zero.o sum_between.o
sum_before_and_after.o -o menu

menu.o: menu.c index_first_zero.h index_last_zero.h sum_between.h
sum_before_and_after.h
     gcc -c -std=c99 menu.c

index_first_zero.o: index_first_zero.c index_first_zero.h
     gcc -c -std=c99 index_first_zero.c

index_last_zero.o: index_last_zero.c index_last_zero.h
     gcc -c -std=c99 index_last_zero.c

sum_between.o: sum_between.c sum_between.h index_first_zero.h
index_last_zero.h
```

```
	gcc -c -std=c99  sum_between.c

sum_before_and_after.o:  sum_before_and_after.c  sum_before_and_after.h
index_first_zero.h index_last_zero.h
	gcc -c -std=c99 sum_before_and_after.c
```