**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. 3388 |  | Трунов Б.Г. |
| Преподаватель |  | Заславский М.М. |

Санкт-Петербург

2023

## Цель работы.

Изучить сборку программ, заголовочные файлы и работу с функциями в Си. Применить полученные знания на практике, реализовав программный код.

## Задание.

Вариант №1

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 20. 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 произведение элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент). (multi\_between\_negative.c) 3 : Найти произведение элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент). (multi\_before\_and\_after\_negative.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

## Выполнение работы.

Файлы:

data\_reading.c — файл содержащий функцию для получения данных из стандартного потока ввода.

data\_reading.h — заголовочный файл с прототипом функции.

index\_first\_negative.c - файл содержащий функцию для получения индекса первого отрицательного числа в массиве.

index\_first\_negative.h — заголовочный файл с прототипом функции.

index\_last\_negative.c - файл содержащий функцию для получения индекса

последнего отрицательного числа в массиве.

index\_last\_negative.h — заголовочный файл с прототипом функции.

menu.c — файл содержащий главную функцию main.

multi\_between\_negative.c - файл содержащий функцию для нахождения произведения элементов массива от первого отрицательного(включительно) до последнего отрицательного (не включительно).

multi\_between\_negative.h — заголовочный файл с прототипом функции.

multi\_before\_and\_after\_negative.c — файл содержащий функцию для произведения элементов массива от начала до первого отрицательного(не включительно) и от последнего отрицательного(включительно) до конца.

multi\_before\_and\_after\_negative.h — заголовочный файл с прототипом функции.

print\_solve.c — файл содержащий функцию для вывода ответа в стандартный поток вывода.

print\_solve.h — заголовочный файл с прототипом функции.

makefile — файл с инструкциями для утилиты make, которая нужна для автоматической сборки проекта.

Подключенные библиотеки:

stdio.h — для реализации ввода/вывода.

Индетификаторы define:

MAX\_SIZE\_ARR 20 - максимальный размер массива, исходя из условия.

Переменные:

int array[MAX\_SIZE\_ARR] — массив чисел из стандартного потока ввода.

int operation — номер операции введенный пользователем.

int array\_count\_items— количество элементов массива array.

int multi\_result — произведений элементов массива функций в функциях multi\_between\_negative, multi\_before\_and\_after\_negative.

Функции:

void data\_reading(int arr[], int \* operation, int \* arr\_count\_items) - принимает массив, указатели на operation и arr\_count\_items. Читает данные из stdin, заполняет массив, разыменовывая operation и arr\_count\_items записывает в них значение.

int index\_first\_negative(int arr[], int arr\_count\_items) — принимает массив, количество элементов массива. Возвращает индекс первого отрицательного элемента в данном массиве.

int index\_last\_negative(int arr[], int arr\_count\_items) — принимает массив, количество элементов массива. Возвращает индекс последнего нуля в данном массиве.

int main() - ничего не принимает. В ней вызываются другие функции, с помощью которых осуществляется основная работа программы.

int multi\_between\_negative(int arr[], int arr\_count\_items) — принимает массив, количество элементов массива. Возвращает произведение элементов массива от первого отрицательного до последнего отрицательного индекса.

int multi\_before\_and\_after\_negative(int arr[], int arr\_count\_items) — принимает массив, количество элементов массива. Возвращает произведение элементов массива от элемента с индексом 0 до первого отрицательного(не включительно) и от последнего отрицательного(включительно) до конца массива.

void print\_solve(int arr[], int operation, int arr\_count\_items) — принимает массив, номер операции, количество элементов массива .В зависимости от номера операции 0,1,2,3 выводит результат соответствующей функции.В случае неизвестного номера операции в стандартный поток вывода stdout выводится «Данные некорректны».

Разработанный программный код см. в приложении А

## Тестирование.

Результаты тестирования представлены в табл. 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1 | 0 -5 -3 -5 -8 3 -9 -3 | 0 | Индекс первого отрицательного числа - 0 |
| 2 | 1 -21 10 0 -23 -7 -15 -14 8 -9 10 -13 -14 -27 0 -7 12 -15 | 16 | Индекс последнего отрицательного числа - 16 |
| 3 | 2 29 1 2 -3 4 5 6 -1 2 1 89 | -360 | Произведение элементов от 3 до 6 индексов - (-360) |
| 4 | 3 29 1 2 -3 4 5 6 -1 1 2 | -116 | Произведение элементов (от 0 до 2) и (от 7 до 9) индексов - -116 |
| 5 | 4 -21 120 0 -23 -7 -15 -14 8 -9 10 -123 -14 -127 0 -7 12 -318 | Данные некорректны | Функции, которой бы соответствовал бы номер 4 не существует |

Таблица 1 – Результаты тестирования.

## Выводы.

Были исследованы: сборка программ, работа с заголовочными файлами, использование функций, написанных лично и импортированных из стандартных библиотек.

Разработана программа, выполняющая считывание с клавиатуры исходных данных и команды пользователя. Для обработки команд пользователя использовался оператор множественного выбора switch.

ПРИЛОЖЕНИЯ А  
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: makefile

all: menu.o data\_reading.o index\_first\_negative.o index\_last\_negative.o print\_solve.o multi\_between\_negative.o multi\_before\_and\_after\_negative.o

gcc menu.o data\_reading.o index\_first\_negative.o index\_last\_negative.o print\_solve.o multi\_between\_negative.o multi\_before\_and\_after\_negative.o -o menu

menu.o: menu.c data\_reading.h print\_solve.h

gcc -c menu.c

data\_reading.o: data\_reading.c data\_reading.h

gcc -c data\_reading.c

print\_solve.o: print\_solve.c print\_solve.h index\_first\_negative.h index\_last\_negative.h

gcc -c print\_solve.c

index\_first\_negative.o: index\_first\_negative.c index\_first\_negative.h

gcc -c index\_first\_negative.c

index\_last\_negative.o: index\_last\_negative.c index\_last\_negative.h

gcc -c index\_last\_negative.c

multi\_between\_negative.o: multi\_between\_negative.c multi\_between\_negative.h index\_first\_negative.h index\_last\_negative.h

gcc -c multi\_between\_negative.c

multi\_before\_and\_after\_negative.o: multi\_before\_and\_after\_negative.c multi\_before\_and\_after\_negative.h index\_first\_negative.h index\_last\_negative.h

gcc -c multi\_before\_and\_after\_negative.c

clean:

rm -f \*.o

rm -f \*.out

Название файла: menu.c

#include "data\_reading.h"

#include "print\_solve.h"

#define MAX\_SIZE\_ARR 20

int main(){

int array[MAX\_SIZE\_ARR];

int operation = -1;

int array\_count\_items = 0;

data\_reading(array, &operation, &array\_count\_items);

print\_solve(array, operation, array\_count\_items);

return 0;

}

Название файла: data\_reading.c

#include <stdio.h>

#include "data\_reading.h"

#define MAX\_SIZE\_ARR 20

void data\_reading(int arr[], int \* operation, int \* arr\_count\_items){

scanf("%d", operation);

do{

if (\*arr\_count\_items < MAX\_SIZE\_ARR){

scanf("%d", &arr[(\*arr\_count\_items)++]);

}

}while(getchar() != '\n');

}

Название файла: data\_reading.h

#ifndef DATA\_READING\_HEADER

#define DATA\_READING\_HEADER

void data\_reading(int arr[], int \* operation, int \* arr\_count\_items);

#endif

Название файла: index\_first\_negative.c

#include "index\_first\_negative.h"

int index\_first\_negative(int arr[], int arr\_count\_items){

for (int x = 0; x < arr\_count\_items; x++){

if (arr[x] < 0){

return x;

break;

}

}

return -1;

}

Название файла: index\_first\_negative.h

#ifndef INDEX\_FIRST\_NEGATIVE\_HEADER

#define INDEX\_FIRST\_NEGATIVE\_HEADER

int index\_first\_negative(int arr[], int arr\_count\_items);

#endif

Название файла: index\_last\_negative.c

#include "index\_last\_negative.h"

int index\_last\_negative(int arr[], int arr\_count\_items){

for (int x = arr\_count\_items - 1; x > 0; x--){

if (arr[x] < 0){

return x;

break;

}

}

return -1;

}

Название файла: index\_last\_negative.h

#ifndef INDEX\_LAST\_NEGATIVE\_HEADER

#define INDEX\_LAST\_NEGATIVE\_HEADER

int index\_last\_negative(int arr[], int arr\_count\_items);

#endif

Название multi\_between\_negative.c

#include "multi\_between\_negative.h"

#include "index\_first\_negative.h"

#include "index\_last\_negative.h"

int multi\_between\_negative(int arr[], int arr\_count\_items){

int first\_negative = index\_first\_negative(arr, arr\_count\_items);

int last\_negative = index\_last\_negative(arr, arr\_count\_items);

int multi\_result = 1;

for (int x = first\_negative; x < last\_negative; x++){

multi\_result \*= arr[x];

}

return multi\_result;

}

Название файла: multi\_between\_negative.h

#ifndef MULTI\_BETWEEN\_NEGATIVE\_HEADER

#define MULTI\_BETWEEN\_NEGATIVE\_HEADER

int multi\_between\_negative(int arr[], int arr\_count\_items);

#endif

Название файла: multi\_before\_and\_after\_negative.c

#include "multi\_before\_and\_after\_negative.h"

#include "index\_first\_negative.h"

#include "index\_last\_negative.h"

int multi\_before\_and\_after\_negative(int arr[], int arr\_count\_items){

int first\_negative = index\_first\_negative(arr, arr\_count\_items);

int last\_negative = index\_last\_negative(arr, arr\_count\_items);

int multi\_result = 1;

for (int x = 0; x < first\_negative; x++){

multi\_result \*= arr[x];

}

for (int x = last\_negative; x < arr\_count\_items; x++){

multi\_result \*= arr[x];

}

return multi\_result;

}

Название файла: multi\_before\_and\_after\_negative.h

#ifndef MULTI\_BEFORE\_AND\_AFTER\_NEGATIVE\_HEADER

#define MULTI\_BEFORE\_AND\_AFTER\_NEGATIVE\_HEADER

int multi\_before\_and\_after\_negative(int arr[], int arr\_count\_items);

#endif

Название файла: print\_solve.c

#include <stdio.h>

#include "print\_solve.h"

#include "index\_first\_negative.h"

#include "index\_last\_negative.h"

#include "multi\_between\_negative.h"

#include «multi\_before\_and\_after\_negative.h"

#define PRINT\_DOUBLE "%d\n"

void print\_solve(int arr[], int operation, int arr\_count\_items){

switch(operation){

case 0:

printf(PRINT\_DOUBLE, index\_first\_negative(arr, arr\_count\_items));

break;

case 1:

printf(PRINT\_DOUBLE, index\_last\_negative(arr, arr\_count\_items));

break;

case 2:

printf(PRINT\_DOUBLE, multi\_between\_negative(arr, arr\_count\_items));

break;

case 3:

printf(PRINT\_DOUBLE, multi\_before\_and\_after\_negative(arr, arr\_count\_items));

break;

default:

printf("Данные некорректны\n");

break;

}

}

Название файла: print\_solve.h

#ifndef PRINT\_SOLVE\_HEADER

#define PRINT\_SOLVE\_HEADER

void print\_solve(int arr[], int operation, int arr\_count\_items);

#endif