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. 3341		Байрам Э.
Преподаватель		Шевелева А.М.

Санкт-Петербург

2023

Цель работы

Цель этого кода - предоставить пользователю функцию меню, которая принимает ввод пользователя в виде одного из вариантов (0, 1, 2 или 3) и массива целых чисел, и затем выполняет определенные операции. Операции и функции, выполняющие их, определены в отдельных файлах (max.c, min.c, diff.c и sum.c). В зависимости от выбора пользователя, выполняются различные операции, такие как нахождение максимального, минимального, разницы между максимальным и минимальным значением или суммы элементов массива до минимального значения. Эти функции вызываются с использованием соответствующего кода из указанных файлов. Таким образом, данный код предоставляет пользователю возможность выбора различных операций и их выполнения.

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".

Выполнение работы

Программа, написанная на языке программирования C, в начале получает от пользователя номер операции от 0 до 3 и массив целых чисел. Затем программа обрабатывает этот массив, используя метод, указанный номером операции, и выводит результат на экран. При этом обеспечивается дружественный интерфейс для пользователя, чтобы ввод был легким и понятным. Результаты операций четко представляются пользователю, и, при необходимости, предоставляются дополнительные сведения или опции для выбора с использованием соответствующих методов коммуникации.

Первый метод (max) анализирует массив чисел и выводит максимальное по модулю число в массиве.

Второй метод (min) анализирует массив чисел и выводит минимальное по модулю число в массиве.

Третий метод (diff) анализирует массив чисел и выводит разницу между максимальным по модулю и минимальным по модулю элементом.

Четвертый метод (sum) анализирует массив чисел и выводит сумму элементов массива, расположенных после максимального по модулю элемента.

Основная функция считывает строку, содержащую номер метода и массив чисел, и в зависимости от числа запускает соответствующую функцию обработки массива через ключевую структуру.

Переменные, используемые в этой программе:

Max_legent: константа, представляющая максимальное изменение массива, используемое в программе.

size_array: отслеживает количество существующих элементов в массиве.

ввод: сумма номера транзакции, которая будет использоваться для транзакций массива.

пробел: проверяется, введена ли программа, введенная пользователем, с правильным интервалом.

max: сохраняет число с наибольшим абсолютным значением.

min: сохраняет число с наименьшим абсолютным значением.

sum_num: хранит сумму, которая будет использоваться в общей транзакции.

arr[]: массив как распределение от основной функции к другим функциям.

int abs_max(int arr[], int size) — эта функция принимает массив и его длину и возвращает число с наибольшим абсолютным значением в массиве.

int abs_min(int arr[], int size) — эта функция принимает массив и его длину и возвращает число с абсолютным наименьшим значением в массиве.

Эта программа иллюстрирует функционирование условий, конструкции switch-case, циклов for и while, а также работу функций с возвращаемыми значениями в языке C. Она демонстрирует использование различных логических и арифметических операций над целыми числами.

Выводы

Этот код предоставляет пример простой консольной программы, в которой пользователь может выбирать разные операции для выполнения над заданным массивом целых чисел. Он демонстрирует организацию кода с использованием отдельных файлов для различных функций, что может улучшить читаемость и обслуживаемость проекта.

Некоторые выводы:

1. Организация кода: Разделение функций по отдельным файлам (max.c, min.c, diff.c и sum.c) способствует легкости обслуживания и расширения проекта.

2. Взаимодействие с пользователем: Программа предоставляет пользователю меню с выбором операций, что делает ее более интерактивной и удобной в использовании.

3. Работа с массивами: Примеры операций (нахождение максимума, минимума, разницы и суммы элементов) демонстрируют работу с массивами целых чисел, что может быть полезно в реальных проектах.

4. Потенциальное улучшение: Этот код может быть доработан, чтобы добавить проверку ввода пользователя и управление ошибками, чтобы сделать программу более надежной.

В целом, это небольшой, но показательный пример кода, который может быть базой для разработки более сложных консольных приложений с использованием меню выбора операций.

Ход работы

1. Создаем цель all, которая будет компилировать все объектные файлы и собирать их в исполняемый файл menu. Для этого используем команду gcc с флагом -std=gnu99 для поддержки стандарта C99.

makefile

```
all: menu.o min.o max.o diff.o sum.o
 gcc -std=gnu99 menu.o min.o max.o diff.o sum.o -o menu
```

2. Создаем цели для каждого объектного файла. Каждая цель компилирует соответствующий исходный файл в объектный файл. Используем команду gcc с флагом -std=gnu99 и опцией -c для компиляции без линковки.

makefile

```
menu.o: menu.c
 gcc -std=gnu99 -c menu.c
min.o: min.c
    gcc -std=gnu99 -c min.c
max.o: max.c
    gcc -std=gnu99 -c max.c
diff.o: diff.c
    gcc -std=gnu99 -c diff.c
sum.o: sum.c
    gcc -std=gnu99 -c sum.c
```

3. Сохраняем файл с расширением .makefile или без расширения, например, Makefile.

4. Запускаем команду make в терминале для компиляции и сборки программы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: menu.c

```
#include <stdio.h>
#include <stdlib.h>
#include "max.h"
#include "min.h"
#include "diff.h"
#include "sum.h"
#define MAX_LENGHT 100

int main()
{
    int array[MAX_LENGHT];
    int input;
    int size_array = 0;
    char space = ' ';

    scanf("%d", &input);
    while (size_array < MAX_LENGHT && space == ' ')
    {
        scanf("%d%c", &array[size_array], &space);
        size_array++;
    }

    switch (input)
    {
    case 0:
        printf("%d\n", max(array, size_array));
        break;
    case 1:
        printf("%d\n", min(array, size_array));
        break;
    case 2:
        printf("%d\n", diff(array, size_array));
        break;
    case 3:
        printf("%d\n", sum(array, size_array));
        break;
    default:
        printf("Данные некорректны\n");
        break;
    }
}
```

Название файла: Makefile

```
all: menu.o min.o max.o diff.o sum.o
 gcc menu.o min.o max.o diff.o sum.o -o menu

menu.o: menu.c min.h max.h diff.h sum.h
 gcc -gnu=std99 -c menu.c
```

```
min.o: min.c
 gcc -gnu=std99 -c min.c

max.o: max.c
 gcc -gnu=std99 -c max.c

diff.o: diff.c
 gcc -gnu=std99 -c diff.c

sum.o: sum.c
 gcc -gnu=std99 -c sum.c
```