**Лабораторна робота No7. Функції**

**1 Вимоги**

**1.1 Розробник**

* Бреславець Михайло Юрійович
* Студент групи: КІТ-121а
* 21-гр-2021

**1.2 Загальне завдання**

Переробити програми, що були розроблені під час виконання лабораторних робіт тем “Масиви” та “Цикли” таким чином, щоб використовувалися функції для обчислення результату. Функції повинні задовольняти основну їх причетність - уникати дублювання коду. Тому, для демонстрації роботи, ваша програма (функція main()) повинна мати можливість викликати розроблену функцію з різними вхідними даними.

**2 Опис програми**

**2.1 Функціональне призначення**

Програма призначена для перевірки числа чи є воно простим та зcува матриці вліво

**2.2 Опис логічної структури**

**Функція визначення простоти числа**

bool is\_simple();

*Призначення*: визначення чи є число прости.

*Опис роботи*: функція у циклі ділить передану змінну на числа від 1 до кореня з цього числа та перевіряє залишок від ділення.

**Аргументи**:

* *input\_number* - число, яке треба перевірити на простоту;

**Функція здвига матриці вліво**

void shift\_matrix(int matrix[], int N, int result\_matrix[]){

*Призначення*: здвиг матриці вліво.

*Опис роботи*: здвигає матрицю вліво з переносом першого елемента у кінець.

**Аргументи**:

* *matrix* - масив, в який записана матриця, яку треба здвинути;
* *N* - розмір матриці.
* *result\_matrix* - масив, в який записується результат.

**Основна функція**

int main()

*Призначення*: головна функція.

*Опис роботи*:

* створення друхвимірної матриці із заданих чисел за допомогою циклів
* здвиг матріці за допомогою функції shift\_matrix
* перевірка числа на складенність за допомогою функції is\_simple

**Структура проекту**

└── lab07

├── Doxyfile

├── Makefile

├── README.md

├── doc

│   └── lab08.md

│   └── lab08.pdf

│   └── lab08.docx

└── src

└── main.c

**3 Варіанти використання**

Для демонстрації результатів кожної задачі використовується:

* покрокове виконання програми в утиліті lldb;
* видача результатів у консоль за допомогою call

**Варіант використання 1**: послідовність дій для запуску програми у режимі відлагодження:

* запустити програму у відлагоднику lldb;
* поставити точку зупинки на функції main (строка з return 43;);
* запустити програму;
* подивитись результати виконання програми за допомогою fr v

misha@misha-ThinkPad-E470:~/Desktop/programming-bresla/lab07$ lldb src/main.bin

(lldb) target create "src/main.bin"

Current executable set to '/home/misha/Desktop/programming-bresla/lab07/src/main.bin' (x86\_64).

(lldb) l1

error: 'l1' is not a valid command.

(lldb) l 1

1 //#include <iostream.h>

2 #include <stdlib.h>

3 #include <stdio.h>

4 #include <stdbool.h>

5 void shift\_matrix();

6 bool is\_simple();

7 int main(){

8 srand(20);

9 int input\_number = 37; //введення числа на перевірку

10 scanf("%d", &input\_number);

(lldb) l

11 bool res = is\_simple(input\_number); //змінна що буде відповідати за результат

12 const int N = 4; // граніци масива

13 int original\_matrix[N][N];

14 int result\_matrix[N\*N];

15 for(int i = 0; i<N; i++){ //заповнювання двувимірного масива рандомними числами

16 for(int j = 0; j < N; j++){

17 original\_matrix[i][j] = rand()%10;

18 }

19 }

20

(lldb) l

21 for(int i = 0; i<N; i++){ //заповнювання двувимірного масива з консолі

22 for(int j = 0; j < N; j++){

23 scanf("%d", &original\_matrix[i][j]);

24 }

25 }

26

27

28 int matrix[N\*N];

29 for(int i = 0; i < N; i++){ // перевод до одновимірного масива

30 for(int j = 0; j < N; j++){

(lldb) l

31 matrix[i \* N + j] = original\_matrix[i][j];

32 }

33 }

34

35 shift\_matrix(matrix, N, result\_matrix);

36

37 int shifted\_matrix[N][N]; //перевод обратно до двувимірного виду

38 for(int j = 0; j < N; j++){

39 for(int k = 0; k < N; k++){

40 shifted\_matrix[j][k] = result\_matrix[j\*N+k];

(lldb) l

41 }

42 }

43 return 0;

44 }

45 void shift\_matrix(int matrix[], int N, int result\_matrix[]){

46 int cur = 0;

47 for(int i = 0; i < N; i++){

48 cur = matrix[i\*N]; // передаю значення першого елемента строки

49 for(int j = 0; j < N; j++){

50 if(j+1 == N){ //перевірка, чи є ций елемент останнім у строці

(lldb) b 43

Breakpoint 1: where = main.bin`main + 591 at main.c:43:2, address = 0x000000000040139f

(lldb) run

Process 1714 launched: '/home/misha/Desktop/programming-bresla/lab07/src/main.bin' (x86\_64)

/

Process 1714 stopped

\* thread #1, name = 'main.bin', stop reason = breakpoint 1.1

frame #0: 0x000000000040139f main.bin`main at main.c:43:2

40 shifted\_matrix[j][k] = result\_matrix[j\*N+k];

41 }

42 }

-> 43 return 0;

44 }

45 void shift\_matrix(int matrix[], int N, int result\_matrix[]){

46 int cur = 0;

(lldb) fr v

(int) input\_number = 37

(bool) res = true

(const int) N = 4

(int [4][4]) original\_matrix = {

[0] = ([0] = 1, [1] = 8, [2] = 7, [3] = 9)

[1] = ([0] = 6, [1] = 1, [2] = 2, [3] = 1)

[2] = ([0] = 0, [1] = 1, [2] = 0, [3] = 4)

[3] = ([0] = 7, [1] = 9, [2] = 4, [3] = 9)

}

(int [16]) result\_matrix = {

[0] = 8

[1] = 7

[2] = 9

[3] = 1

[4] = 1

[5] = 2

[6] = 1

[7] = 6

[8] = 1

[9] = 0

[10] = 4

[11] = 0

[12] = 9

[13] = 4

[14] = 9

[15] = 7

}

(int [16]) matrix = {

[0] = 1

[1] = 8

[2] = 7

[3] = 9

[4] = 6

[5] = 1

[6] = 2

[7] = 1

[8] = 0

[9] = 1

[10] = 0

[11] = 4

[12] = 7

[13] = 9

[14] = 4

[15] = 9

}

(int [4][4]) shifted\_matrix = {

[0] = ([0] = 8, [1] = 7, [2] = 9, [3] = 1)

[1] = ([0] = 1, [1] = 2, [2] = 1, [3] = 6)

[2] = ([0] = 1, [1] = 0, [2] = 4, [3] = 0)

[3] = ([0] = 9, [1] = 4, [2] = 9, [3] = 7)

}

**Висновки**

На цій лабораторній роботі я переніс дві минулі програми в функції, щоб уникнути копіювання коду та продемонстрував роботу.