

# Лабораторна робота №8

**Тема:** Вступ до блок-схем алгоритмів

**Розробник:** Макаренко Владислав Олександрович

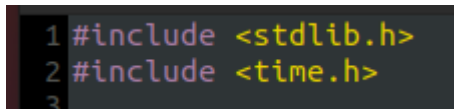
**Перевіряючий:** Челак Віктор Володимирович

**Загальне завдання:** Для кожної розробленої функції, що були виконані у попередній роботі, слід зробити схему алгоритмів.

## Опис програми

1. За допомогою команди «папо» відкриваємо текстовий редактор та розпочинаємо писати код нашої програми.

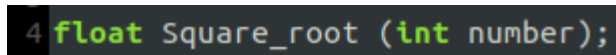
2. Підключаємо бібліотеки для функції «rand()»(Рис.1).



```
1 #include <stdlib.h>
2 #include <time.h>
3
```

Рисунок 1 – необхідні бібліотеки

3. Оголошуємо функцію(Рис.2).



```
4 float Square_root (int number);
```

Рисунок 2 – функція

4. В функції «main» оголошимо змінну «number» для числа, з якого буде братись корінь, та двічі викликаємо функцію «Square\_root». В першому випадку функція буде розраховувати корінь для випадкового числа, згенерованого функцією «rand()»; в другому випадку функція розраховує корінь

з заданого нами числа. Та результат записує відповідно в змінні «result1» та «result2» (Рис.3). Блок-схема алгоритму функції «main» подана на рис.4.

```
6 int main ()
7 {
8     srand(time(NULL));
9     int number;
10    float result1 = Square_root (number = rand() % 50);
11    float result2 = Square_root (number = 25);
12    return 0;
13 }
14
```

Рисунок 3 – функція main()

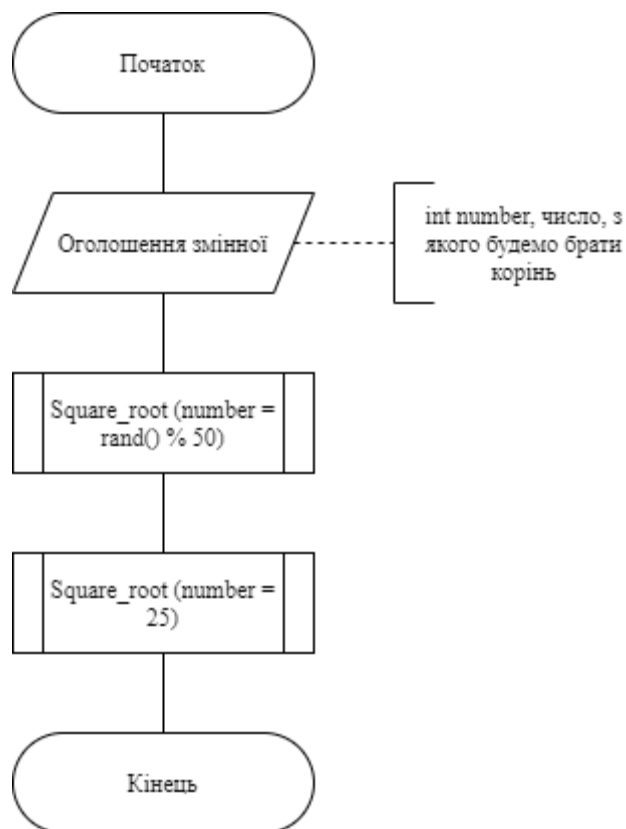


Рисунок 4 – схема алгоритму функції main

5. Описуємо функцію для розрахунку квадратного кореня з числа. В якій за допомогою циклу «while» збільшуємо значення «result» до тих пір поки воно помножене на себе не дасть нам наше початкове число «number» (Рис.5). Блок-схема алгоритму функції «Square\_root» подана на рис.6.

```

float Square_root (int number)
{
    float result = 0.0001;
    while (result * result <= number) {
        result += 0.0001;
    }

    return result;
}

```

Рисунок 5 – функція Square\_root

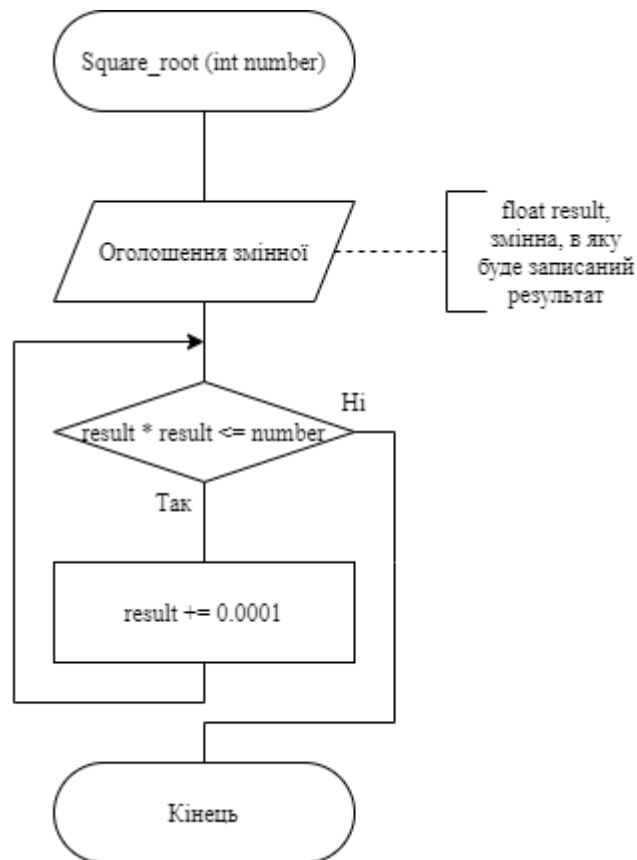


Рисунок 6 – схема алгоритму функції Square\_root

6. Зберігаємо текстовий файл під назвою «main1.c» та компілюємо проект за допомогою команди «make clean prep compile».
7. За допомогою відлагодника «nemiver» демонструємо роботу програми (Рис.7).

The screenshot shows a debugger window titled "main1.bin (путь=«/home/vlad/Programing-repo/lab07/dist/main1.bin», pid=2781) - Nemiver". The menu bar includes "Файл", "Правка", "Вид", "Отладка", and "Помощь". The toolbar has buttons for "Продолжить", "Отладка", "Запустить или перезапустить", and "Остановить". The main window displays the source code of "main1.c" with the following content:

```
1 #include <stdlib.h>
2 #include <time.h>
3
4 float Square_root (int number); // задамо функцію для розрахунку квадратного кореня з задано числа
5
6 int main ()
7 {
8     srand(time(NULL));
9     int number;
10    float result1 = Square_root (number = rand() % 50); //робимо перший виклик для випадкового чи
11    float result2 = Square_root (number = 25); //робимо другий виклик для заданого нами числа
12    return 0;
13 }
14
15 //описуємо функцію для розрахунку квадратного кореня
16 float Square_root (int number)
17 {
18     float result = 0.0001;
19     while (result * result <= number) {
20         result += 0.0001;
21     }
22     return result;
23 }
24
25
```

The status bar at the bottom right indicates "Строка: 25, Столбец: 1". Below the code editor, there is a table showing the current state of variables:

Переменная	Значение	Тип
Локальные переменные		
number	16	int
result1	4.00006485	float
result2	5.00002098	float
Параметры функции		

The bottom of the window features a tabbed interface with the following tabs: "Терминал цели", "Контекст", "Точки останова", "Регистры", "Память", and "Монитор выражений". The "Контекст" tab is currently selected.

Рисунок 7 – Демонстрація програми

## Висновки

Ми навчилися створювати блок-схеми алгоритмів розроблених програм та функцій.