

# **Лабораторна робота №10. Документація проекту**

## **1. Вимоги**

### **1.1 Розробник**

Макаренко Владислав Олександрович;

студент групи КІТ-120а;

16-груд-2020

### **1.2 Загальне завдання**

Розробити повноцінний звіт для лабораторної роботи “Функції”

### **1.3 Індивідуальне завдання**

За допомогою функцій отримати корінь заданого числа.

## **2. Опис програми**

### **2.1 Функціональне призначення**

Програма добуває квадратний корінь з числа за допомогою функції `square_root()`. Результат зберігається у змінній `result`. Демонстрація результату передбачає покрокове виконання програми.

## 2.2 Опис логічної структури програми

Для визначення квадратного кореня з числа викликаємо функцію `Square_root()`, яка приймає параметр: число `number` з якого буде добуватись квадратний корінь. Функція збільшує значення параметру `result` на `0.0001` поки значення `result * result` не буде дорівнювати заданому `number`.

Головна функція `main()`. Приймає задане нами число. Задає випадкове число від 1 до 50. Двічі викликає функцію `Square_root`. Схема алгоритму функції подана на рис. 1.

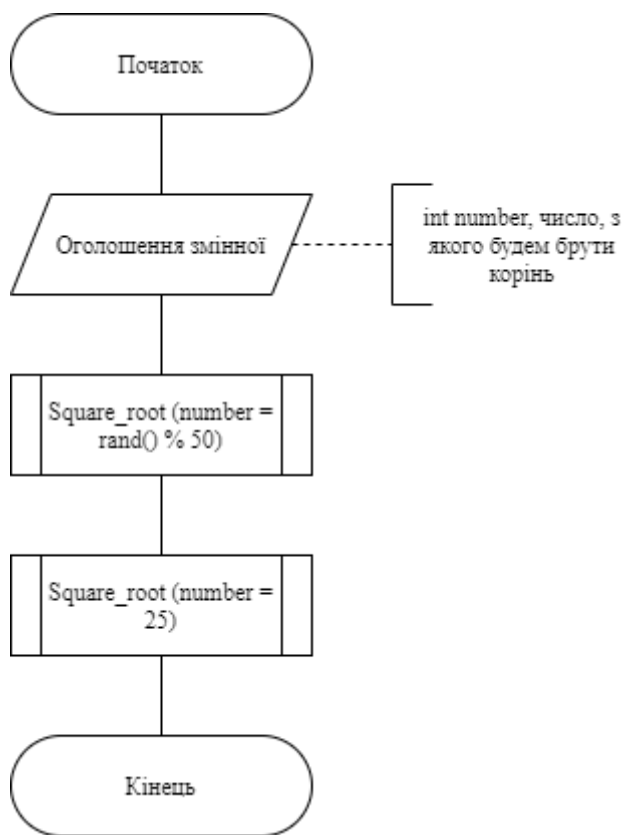


Рисунок 1 — Схема алгоритму функції `main`

Функція, що добуває корінь з числа `Square_root`. Добуває квадратний корінь з числа. Параметри: `number` — число, з якого потрібно добути корінь; `result` — добутий корінь з заданого числа. Функція повертає `result`. Схема алгоритму функції подана на рис. 2.

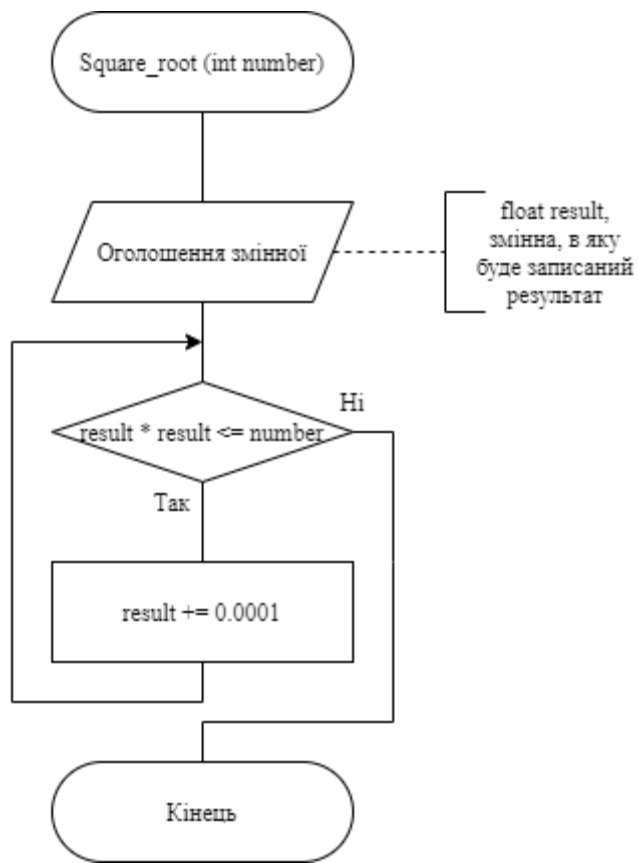


Рисунок 2 — Схема алгоритму функції Square\_rot

## Структура проекту

```

├─lab10
├─ Doxyfile
├─ Makefile
├─ doc
│   ├── lab10.md
│   ├── lab10.docx
│   ├── lab10.pdf
│   └─ assets
│       ├── Square_root().png
│       ├── main ().png
│       ├── main.c.png
│       ├── result1.png
│       └─ result2.png
└─ src
    └─ main.c
  
```

## 2.3 Важливі фрагменти програми

**Підключення заголовочного `stdlib.h` та `time.h` для генерації випадкових чисел.**

```
#include <stdlib.h>
#include <time.h>
```

**Виклик функції для випадкового числа**

```
result1 = Square_root (number = rand() % 50);
```

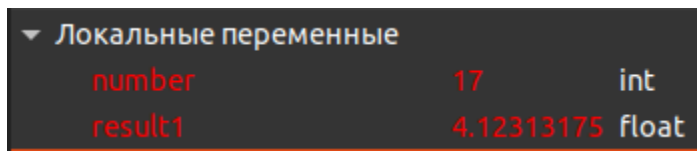
**Добування кореню за допомогою цикла**

```
while (result * result <= number) {
    result += 0.0001;
}
```

## 3. Варіанти використання

Для демонстрації результатів використовується покрокове виконання програми та інші засоби налагодження відладчика `netiver`. Нижче наводиться послідовність дій запуску програми у режимі відладження.

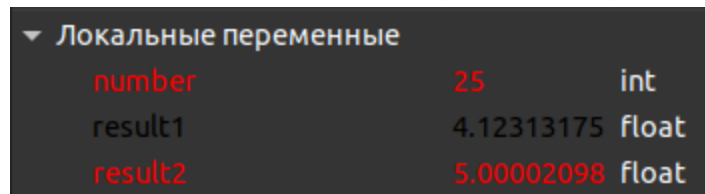
*Крок 1* (рис. 3). Виконуємо функцію для випадково згенерованого числа



Локальные переменные		
<code>number</code>	17	int
<code>result1</code>	4.12313175	float

Рисунок 3 — Результат виконання функції для першого кроку

*Крок 2* (див. рис. 4). Виконуємо функцію для заданого нами числа



▼ Локальные переменные		
number	25	int
result1	4.12313175	float
result2	5.00002098	float

Рисунок 4 — Результат виконання функції для другого кроку

## Висновки

Ми навчилися документувати проект за допомогою Markdown та в doc форматі, згідно ДСТУ.