

# Лабораторна робота №7. Функції

**Автор:** Стась Артем

**Група:** КН-922Б

## **Завдання:**

**1.**Переробити програми, що були розроблені під час виконання лабораторних робіт з тем "Масиви" та "Цикли" таким чином, щоб використовувалися функції для обчислення результату.

**2.**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.

**3.**Слід звернути увагу: параметри одного з викликів функції повинні бути згенеровані за допомогою генератора псевдовипадкових чисел `random()`.

**4.**Слід звернути увагу (#2): продемонструвати встановлення вхідних даних через аргументи додатка (параметри командної строки).

Обробити випадок, коли дані не передались - у цьому випадку вони матимуть значення за умовчуванням, обраними розробником.

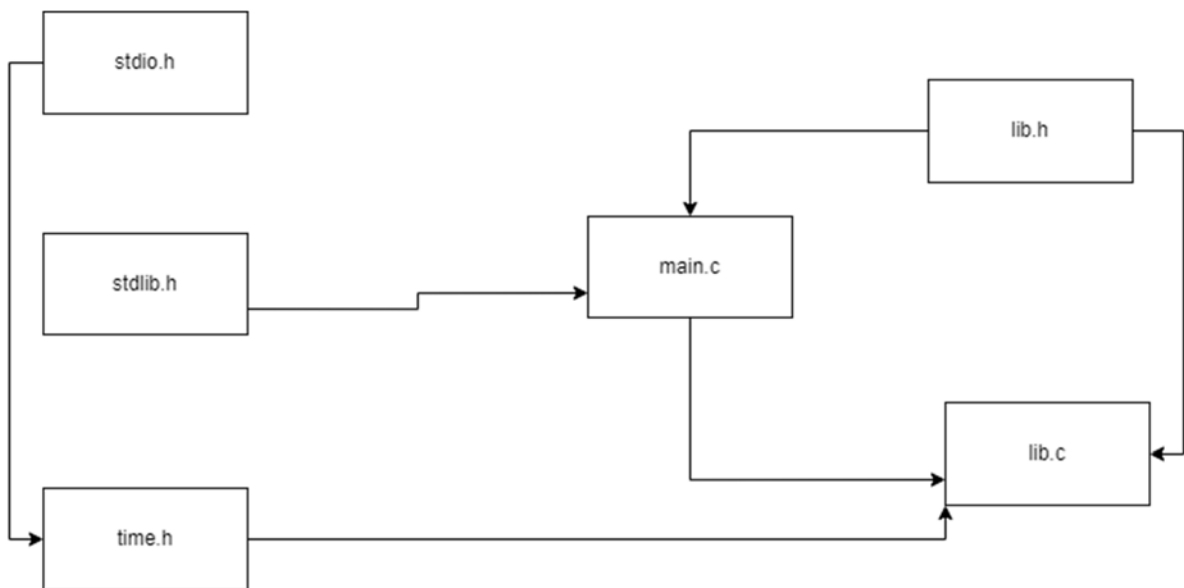
# Опис програми

## Функціональне призначення

Ця програма виконує дві операції.

1. Розташовує числа за методом “бульбашки “
2. Знаходить факторіал заданого числа .

## Опис логічної структури



(мал.1) графічна структура

## Вміст файлу "main.c"

### Головний файл

Це файл, який містить точку входу, виклики функцій lab05, lab06 та значення для аргументів цих функцій.

```
int main(int argc, char *argv[])
```

### Головна функція.

### Послідовність дій

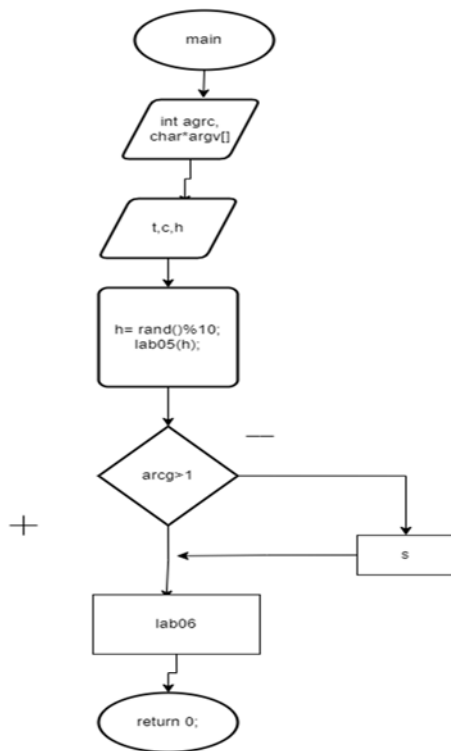
- Присвоїти значення аргументам argc і argv.
  1. argc - аргумент типу int необхідний для обчислення чисел в масиві .
  2. argv - масив типу char який зберігатиме в собі значення від якого почнеться перевірка всіх чисел .
- Створення змінних яким буде надано значення для аргументів функцій link lab05, lab06.
  1. t - зберігає значення за умовчуванням від якого буде починатися перевірка квитків для функції lab06. Воно буде використовуватися якщо не будуть передані аргументи командного рядка.
  2. c - змінна яка використовується для перевірки аргументів командного рядка.
- Генеруємо випадкові числа за допомогою генератора rand() та функції srand(), після чого привласнюємо їх змінним a .
- Потім викликаємо функцію lab05 і присвоюємо їй аргументам значення змінних a.

```
o  srand((unsigned int)time(NULL));  
o  h = rand() % 10;  
o  lab05(h);
```

- Робимо перевірку на те, чи були введені аргументи через командний рядок.
- Якщо одна з перевірок не була пройдена, то присвоюємо аргументам функції lab06 значення за замовчуванням.

```
    lab06(t,s);  
  
}  
else  
{  
    lab06(d,s);  
}  
  
return 0;  
}
```

(мал.2)Схема функції main



### ***Вміст файлу "lib.c"***

Бібліотечний файл

Цей файл містить реалізацію функцій lab05, lab06.

```
int lab05(h)
```

В цю функцію передається рандомне число.

Послідовність дій

- Створення змінної i.

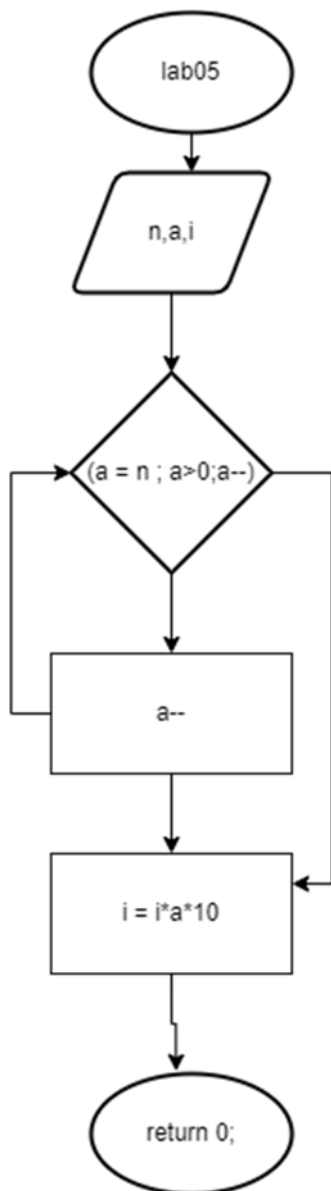
1.  $i$  - використовується для змінна " $i$ " виступає зберігачем добутка самої себе на, ітераційно змінну, змінну " $a$ "

- Запуск циклу у встановленому діапазоні.

- `for (a = h; a>0 ; a--)`

- Розрахунок факторіала за допомогою формул .

- `i= i*a;`



(мал.3) Схема алгоритму функції `lab05`

```
int lab06(long int b[], int s)
```

Аргументи

$i$  - змінна для цикла

j- Змінна для цикла

s- Змінна для виводу даних

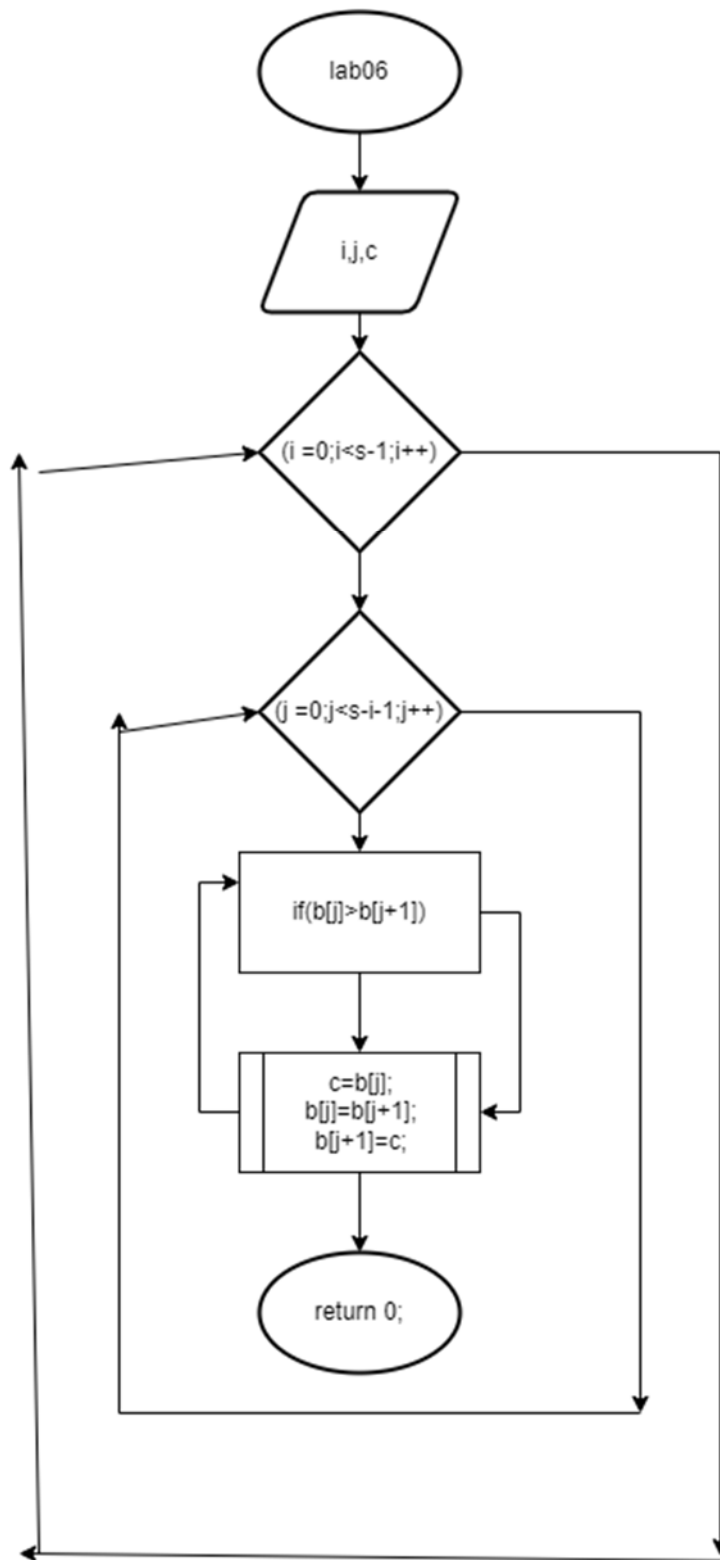
### Послідовність дій

- Створення змінних
- Створення циклу який порівнює два сусідніх елементи.

```
• for ( i = 0; i < s - 1; i++)  
• {  
•     for ( j = 0; j < s - i - 1; j++)  
•     {
```

- якщо вони йдуть у неправильному порядку, то міняємо їх місцями.

```
• if (b[j] > b[j+1])  
• {  
•     c = b[j];  
•     b[j] = b[j+1];  
•     b[j+1] = c;  
• }
```



(мал. 4) Схема алгоритму функції lab06

**Вміст файлу "lib.c"**

Бібліотечний файл



Цей файл містить декларацію функцій lab05, lab06.

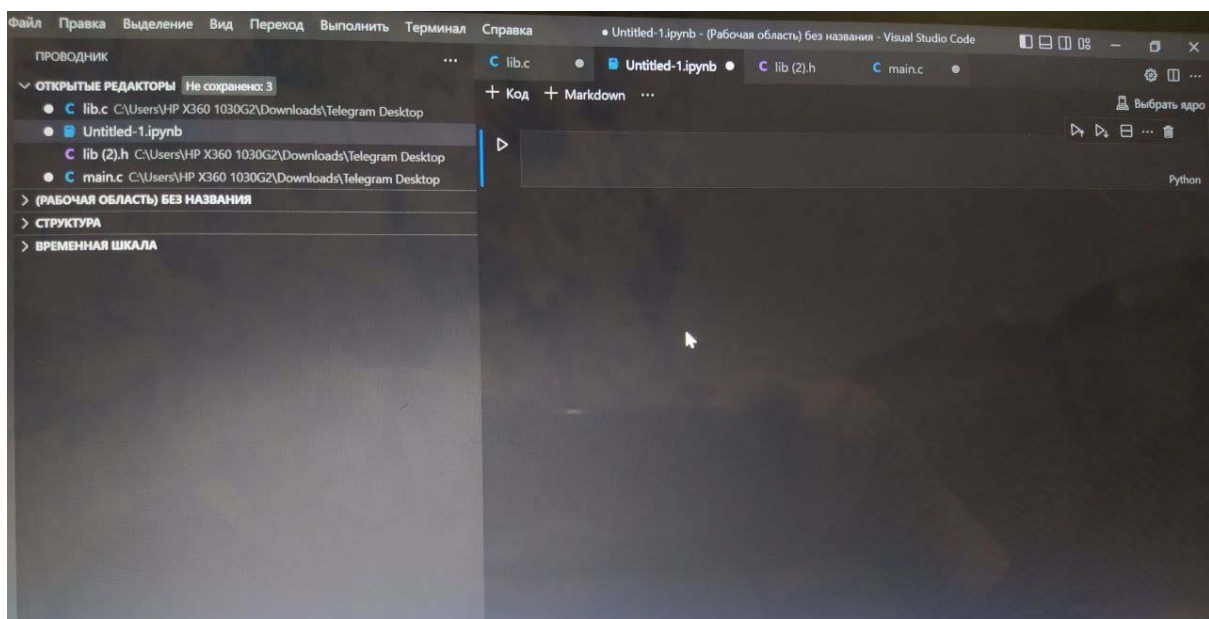
```
int lab05(int h);  
int lab06(long int arr[], unsigned int s );
```

### *Структура проекту лабораторної роботи:*

- |— lab07
- |— Makefile
- |— README.md
- |— src
  - |— lib.c
  - |— lib.h
  - |— main.c

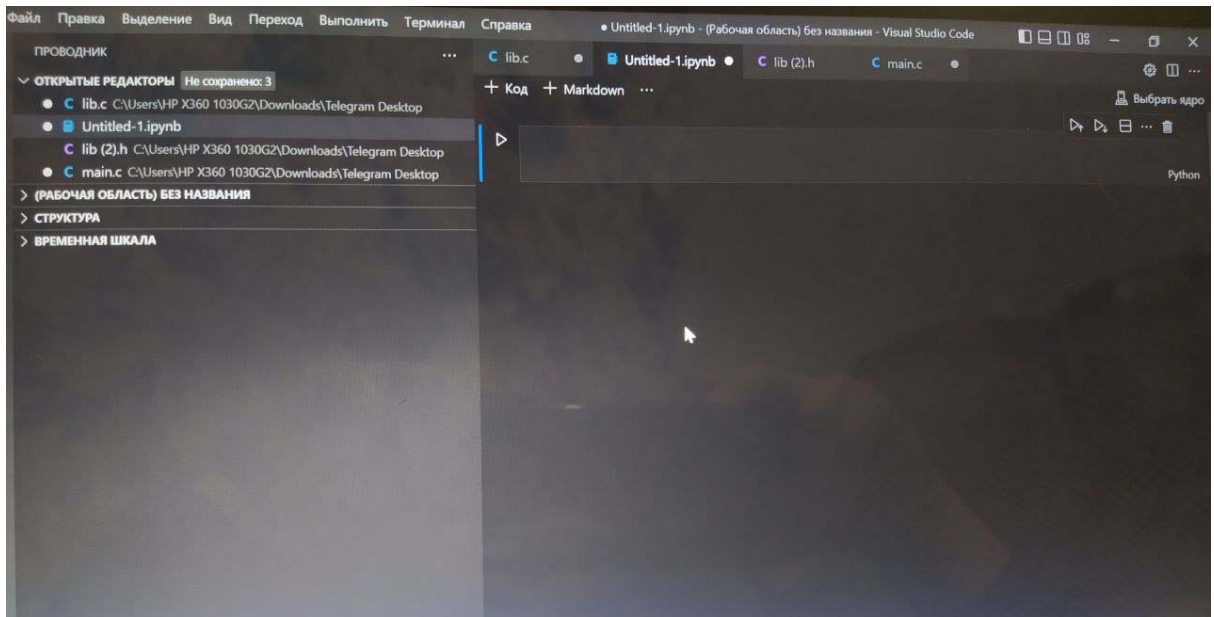
### *Варіанти використання*

1. Ви можете використовувати цю програму двома способами. Перший спосіб-це розрахунок факторіалу числа заданого рандомно (мал. 5).



(мал.5) Як користуватися програмою

2. Другий метод використання цієї програми – це розташування цифр за методом “бульбашки”



(мал6) Як користуватися програмою

3. Щоб побачити результати роботи програми, вам потрібно завантажити її в LLDB, завантажуючи програму. факторіал числа вибереться рандомно . Якщо треба побачити як розташуються числа за методом “бульбашки”, то для цього вам знадобиться точку зупинки для рядка з "return 0"; У функції lab05, та вивести значення (мал5). Якщо ви хочете факторіал числа вибереться рандомно, вам потрібно зробити точку зупинки для рядка з "return 0;" у функції lab06, після того, щоб дізнатися число , вам потрібно вивести змінну i, і щоб переглянути розташування чисел , вам потрібно вивести масив b[](мал 8)



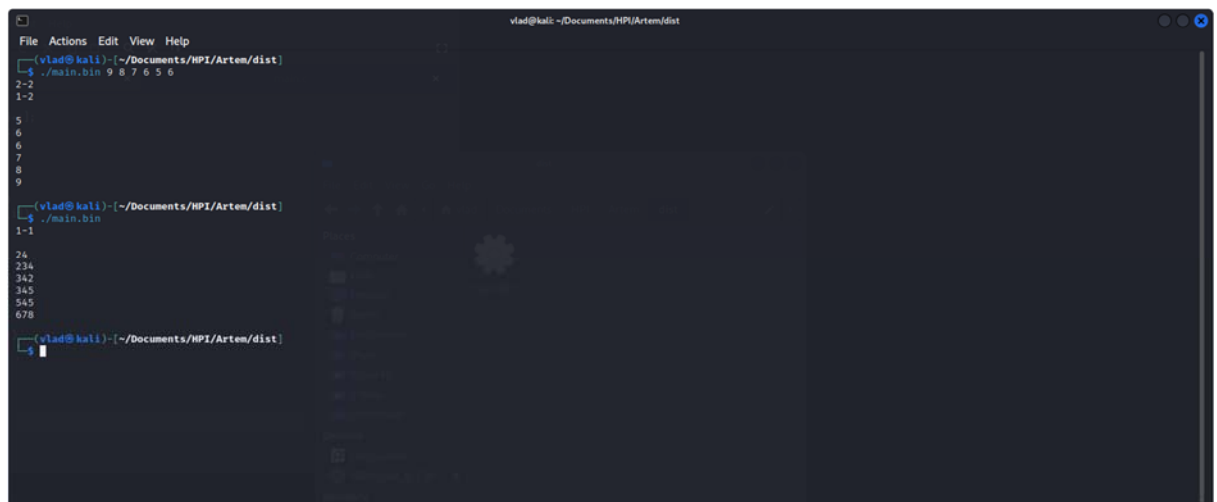
```
File Actions Edit View Help
(vlad@kali)~/.Documents/HPI/Artem/dist
$ ./main.bin
1-1

(vlad@kali)~/.Documents/HPI/Artem/dist
$
.: not enough arguments

(vlad@kali)~/.Documents/HPI/Artem/dist
$ ./main.bin
7-7
6-42
5-210
4-840
3-2520
2-5040
1-5040

(vlad@kali)~/.Documents/HPI/Artem/dist
$
```

(мал. 7) Як дізнатися факторіал числа



```
File Actions Edit View Help
(vlad@kali)~/.Documents/HPI/Artem/dist
$ ./main.bin 9 8 7 6 5 6
1-2
5
6
7
8
9

(vlad@kali)~/.Documents/HPI/Artem/dist
$ ./main.bin
1-1
24
234
342
345
545
678

(vlad@kali)~/.Documents/HPI/Artem/dist
$
```

(мал. 8) Як дізнатися розташування чисел та їх перегляд!

Висновки: у цій роботі з було перетворено лабораторні проекти № 5 та № 6 для використання функцій. Було набуто навичок роботи з функціями, їх декларація, реалізація та виклик. Були отримані, також навички роботи з

бібліотечними файлами їх зв'язком у виконанні різних дій з функціями та їх зв'язком між собою. Під час тестування програми були отримані результати функції lab05 - це отримання факторіала числа та роботи циклу lab06 - це розташування чисел за методом “бульбашки” та їх перегляду.