

## Завдання №1

Це завдання демонструє базове створення та роботу з потоками.

Оригінальний і дочірній потоки по черзі виводять рядки з лічильником і роблять паузи між ітераціями, імітуючи одночасне виконання. Це дозволяє дослідити, як параметри часу впливають на чергування потоків.

```
mint@asus-mint:~/git/op-course/lab5/task1$ ./bin/child_thread
Main Thread. Iteration: 1
Child Thread. Iteration: 1
Main Thread. Iteration: 2
Child Thread. Iteration: 2
Main Thread. Iteration: 3
Child Thread. Iteration: 3
Main Thread. Iteration: 4
Child Thread. Iteration: 4
Child Thread. Iteration: 5
Main Thread. Iteration: 5
Main Thread. Iteration: 6
Child Thread. Iteration: 6
Main Thread. Iteration: 7
Main Thread. Iteration: 8
Child Thread. Iteration: 7
Child Thread. Iteration: 8
Main Thread. Iteration: 9
Child Thread. Iteration: 9
Child Thread. Iteration: 10
Main Thread. Iteration: 10
Both threads completed.
mint@asus-mint:~/git/op-course/lab5/task1$
```

## Завдання №2

У цьому завданні показано, як передавати параметри в потоки. Чотири потоки отримують різні набори аргументів і формують рядки для виведення за певним шаблоном, демонструючи гнучкість параметризації потоків і організацію їх виконання.

```
mint@asus-mint:~/git/op-course/lab5/task2$ ./bin/main
Thread2. World 1
Thread3. Multithreading 1
Thread1. Hello 1
Thread4. Example 1
Thread3. Multithreading 2
Thread2. World 2
Thread1. Hello 2
Thread4. Example 2
Thread2. World 3
Thread1. Hello 3
Thread4. Example 3
Thread2. World 4
Thread4. Example 4
Thread2. World 5
All threads completed.
mint@asus-mint:~/git/op-course/lab5/task2$
```

### Завдання №3

Це завдання зосереджено на синхронізації потоків, де батьківський потік чекає на завершення двох дочірніх потоків. Один з потоків генерує псевдовипадкові числа до появи заданого числа, а інший виконує циклічне виведення тексту; це ілюструє взаємодію потоків і механізми їх завершення.

```
mint@asus-mint:~/git/op-course$ ./lab5/task3/bin/main "Hello, Thread!" 10 1 10
String Thread: Hello, Thread! (iteration 1)
Random Thread: generated number 9
String Thread: Hello, Thread! (iteration 2)
String Thread: Hello, Thread! (iteration 3)
String Thread: Hello, Thread! (iteration 4)
String Thread: Hello, Thread! (iteration 5)
String Thread: Hello, Thread! (iteration 6)
String Thread: Hello, Thread! (iteration 7)
String Thread: Hello, Thread! (iteration 8)
String Thread: Hello, Thread! (iteration 9)
String Thread: Hello, Thread! (iteration 10)
All threads completed. Main thread exiting.
mint@asus-mint:~/git/op-course$
```

## Завдання №4

Завдання демонструє різні способи повернення результатів обчислень із потоків: через глобальні змінні, змінну результату або аргумент. Один потік обчислює числа Каталана, а інший — прості числа, і результати передаються в основний потік, який виводить їх після завершення обчислень.

```
mint@asus-mint:~/git/op-course/lab5/task4$ ./bin/main
Catalan numbers (global variable): 1 1 2 5 14 42 132 429 1430 4862
Prime numbers (global variable): 2 3 5 7 11 13 17 19 23 29
Catalan numbers (return value): 1 1 2 5 14 42 132 429 1430 4862
Prime numbers (return value): 2 3 5 7 11 13 17 19 23 29
Catalan numbers (arg pass): 1 1 2 5 14 42 132 429 1430 4862
Prime numbers (arg pass): 2 3 5 7 11 13 17 19 23 29
```