Лабораторна робота №1

Операційні системи

Студент групи ІПС-32

Чуйко Данило

**Постановка задачі**

I. Study modern os deadlock simulator. Read the user guide. Answer the questions.

II. Answer instructor questions.

III. Realize one of the following

1. Simulate deadlock for 5 dining philosophers.
2. Demonstrate deadlock for incorrect solution for dining philosopher problem.
3. Demonstrate starvation in solution for dining philosopher problem with semaphores.

IV. Consider implementing deadlock avoidance strategy for simulator

**Реалізація**

Константа **NUM\_PHILOSOPHERS**

* Визначає кількість філософів і, відповідно, кількість "палиць".



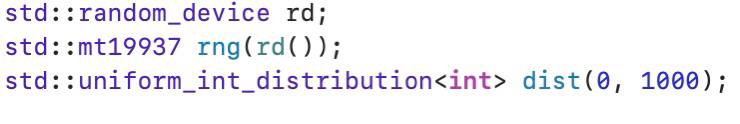
Масив м'ютексів **chopsticks**

* Кожен м'ютекс представляє одну палицю.
* Філософ повинен заволодіти двома сусідніми палицями (м'ютексами) для того, щоб їсти.



Генератор випадкових чисел

* **std::random\_device** і **std::mt19937** використовуються для створення генератора випадкових чисел.
* **std::uniform\_int\_distribution** задає інтервал для випадкових затримок (0-1000 мс), щоб моделювати час, який філософ витрачає на "думання" чи "їжу".

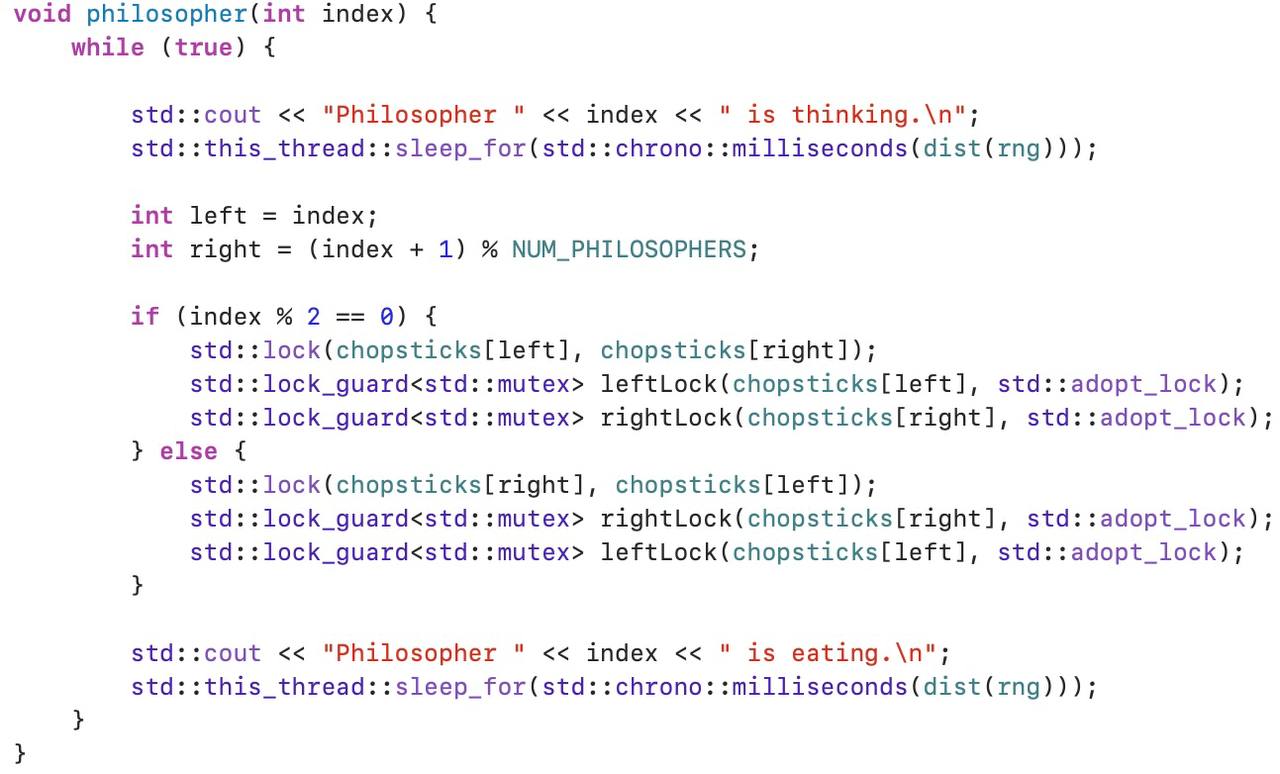


Функція **philosopher**

* Це основна функція, яку виконує кожен потік.
* Думання: Філософ "думає" протягом випадкового часу. Це симулюється затримкою за допомогою **std::this\_thread::sleep\_for**.
* Блокування палиць:

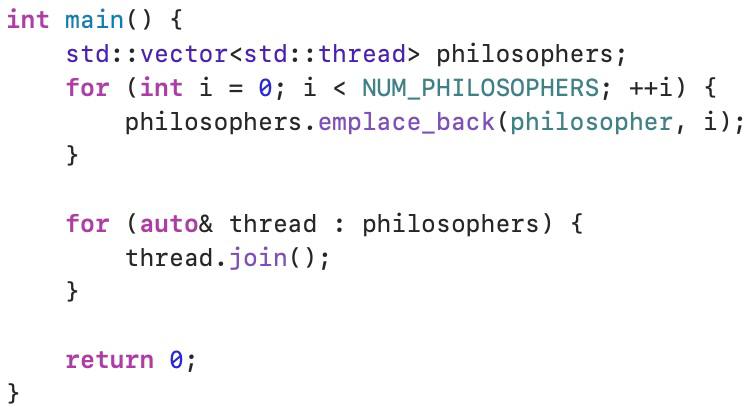
1. Використовується асиметрична стратегія для уникнення deadlock (мертвого блокування).
2. Філософи з парними індексами спочатку блокують ліву палицю, а потім праву. Непарні роблять навпаки.
3. **std::lock** гарантує атомарне блокування обох м'ютексів, уникаючи можливих умов змагання.
4. **std::lock\_guard** забезпечує автоматичне розблокування м'ютексів, коли вони виходять за межі області видимості.

* Їжа: Після захоплення обох палиць філософ "їсть" (затримка), а потім звільняє палиці.



Функція **main**

* Створення потоків: Кожен філософ запускається в окремому потоці.
* Очікування завершення: Потоки працюють нескінченно (**while (true)**), тому програма теоретично ніколи не завершиться, якщо не буде примусово зупинена.



Асиметрія для уникнення мертвого блокування:

Без цієї техніки можливе ситуація, коли всі філософи одночасно захоплять одну палицю і чекатимуть на другу, створюючи deadlock. Використання асиметричного блокування гарантує, що такої ситуації не виникне.

Результат:

