**MODERN C++**

1. **Coroutine**

* Coroutine là một hàm có thể tạm dừng thực thi và tiếp tục bất kì lúc nào.
* Coroutine tạm dừng hàm bằng cách trả về hàm gọi và dữ liệu cần thiết để tiếp tục thực thi được lưu trữ ở một ngăn xếp khác. Điều này giúp thực thi các mã tuần tự một cách bất đồng bộ.
* Một hàm là Coroutine nếu nó chứa bất kì biểu thức sau:
  + - **co\_await**: Để tạm dừng cho đến khi tiếp tục
    - **co\_yield**: Để tạm dừng và trả về 1 giá trị
    - **co\_return**: Để hoàn tất việc thực thi và trả về 1 giá trị
  + 1 Coroutine liên kết với những thành phần sau:
    - **promise object:** Thao tác bên trong coroutine, dùng để coroutine submit kết quả hoặc ngoại lệ
    - **coroutine handle:** Thao tác bên ngoài coroutine, cho phép tương tác với một coroutine đã được khởi tạo. coroutine handle chỉ là một tham chiếu đến coroutine và không trực tiếp quản lý bộ nhớ của nó. Được sử dụng để tiếp tục thực thi coroutine hoặc để huỷ coroutine.
    - **coroutine state:** Là 1 object chứa promise object, các tham số, biểu diễn của điểm tạm dừng hiện tại để biết được cần tiếp tục ở đâu khi resume và các biến cục bộ trong phạm vi khi xoá, các biến cục bộ và biến tạm thời có vòng đời nằm trong điểm tạm dừng.
  + Khi 1 coroutine bắt đầu thực thi thì sẽ thực hiện những điều sau:
    - Cấp phát coroutine state bằng toán tử new
    - Sao chép tất cả tham số hàm vào coroutine state
    - Gọi hàm khởi tạo cho promise object, nếu promise object có 1 hàm khởi tạo với tất cả tham số của coroutine thì hàm khởi tạo đó sẽ được gọi, nếu không thì gọi hàm mặc định.
    - Gọi promise.get\_return\_object() và giữ kết quả trong 1 biến cục bộ. Kết quả của lần gọi đó sẽ được gửi lại cho đối tượng gọi khi coroutine lần đầu đuợc tạm dừng.
    - Gọi promise.initial\_suspend() và co\_awaits kết quả của nó. Các Promise types phổ biến thường trả về std::suspend\_always cho lazily-started coroutines hoặc std::suspend\_never cho early-started coroutines. (Lazily started coroutines là coroutine được tạm dừng ngay khi được gọi khởi tạo, dòng đầu tiên của coroutine body chỉ được gọi khi nó được resume. Early-started coroutines là coroutine được khởi động ngay lập tức khi được gọi)
    - Khi co\_await promise.initial\_suspend() được tiếp tục, bắt đầu thực thi phần body của coroutine.
  + Khi coroutine đến 1 điểm dừng.
    - Return object đạt được lúc trước sẽ được trả về cho đối tượng gọi/tiếp tục.
  + Khi coroutine đến 1 co\_return, nó thực hiện như sau:
    - Gọi promise.return\_void() cho co\_return; co\_return expr; khi expr có kiểu void
    - Hoặc gọi promise.return\_value(expr) cho co\_return expr; khi expr có kiểu khác void.
    - Huỷ tất cả các biến theo trình tự ngược với trình tự mà các biến đó được tạo ra.
    - Gọi promise.final\_suspend() và co\_awaits kết quả.
  + Nếu coroutine kết thúc bằng 1 ngoại lệ chưa được bắt từ trước:
    - Bắt ngoại lệ đó và gọi promise.unhandled\_exception() trong catch-block
    - Gọi promise.final\_suspend() và co\_awaits kết quả.
  + Khi coroutine state bị huỷ vì bị chấm dứt thông qua co\_return hoặc ngoại lệ chưa được bắt, hoặc bị huỷ thông qua coroutine handle nó sẽ thực hiện những việc sau:
    - Gọi destructor từ promise object.
    - Gọi destructor của các bản sao tham số của hàm.
    - Gọi operator delete để giải phóng bộ nhớ được sử dụng bởi trạng thái coroutine.
    - Chuyển thực thi trở lại cho người gọi/tiếp tục.