## Phân biệt và so sánh ưu nhược điểm của các kỹ thuật đồng bộ thread trên Windows: Critical Section, Mutex, Semaphore, Event.

Đây đều là các phương pháp để đồng bộ hóa và loại trừ lẫn nhau của các process và thread

* Critical Section: là một cơ chế “khóa” đơn giản để triển khai và thực thi vùng code quan trọng
  + Ưu điểm:
    - Không phải đối tượng lõi
    - Được duy trì trong không gian của người dùng
    - Tốc độ nhanh
  + Nhược điểm
    - Không thể ra tín hiệu cho thread khác
    - Không có giới hạn thời gian
* Mutex: là một cơ chế khóa tương tự CRITICAL\_SECTION nhưng có nhiều cải tiến
  + Ưu điểm
    - Có giới hạn thời gian và có thể ra tín hiệu khi thread bị hủy bỏ
  + Nhược điểm
* Semaphore: là một cơ chể ra tín hiệu. Semaphore duy trì một số đếm, khi số này > 0 thì sẽ ra tín hiệu, không ra tín hiệu khi = 0
  + Ưu điểm:
    - Cho phép nhiều thread truy cập vào Critical Section
    - Không tốn tài nguyên khi chờ
  + Nhược điểm
    - Có thể khó thực thi
* Event:
  + Ưu điểm:
    - Có thể ra tín hiệu cho nhiều thread

Cả 4 phương pháp trên, nếu không dung đúng cách, có thể gây ra vấn đề như race conditions, deadlocks… và các lỗi khó phát hiện.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **CRITICAL\_SECTION** | **Mutex** | **Semaphore** | **Event** |
| **Được đặt tên, có bảo mật** | Không | Có | Có | Có |
| **Có thể truy cập từ nhiều process** | Không | Có | Có | Có |
| **Cách đồng bộ hóa** | Vào | Chờ | Chờ | Chờ |
| **Cách ra tín hiệu, nhả** | Rời khỏi | Nhả hoặc bỏ | Bất kì thread nào cũng có thể nhả | Set, pulse |
| **Quyền sở hữu** | 1 thread. Thread sở hữu có thể vào nhiều lần. | 1 thread. Thread sở hữu có thể chờ nhiều lần | Không. Nhiều thread 1 lúc, có thể lên đến mức tối đa | Không. Bất kỳ thread nào cũng có thể set, pulse |
| **Sau khi nhả** | 1 thread chờ khác có thể vào | 1 thread chờ khác có thể lấy quyền sở hữu | Nhiều thread có thể thực hiện, phụ thuộc vào số đếm | 1 hoặc nhiều thread chờ có thể thực hiện |