1. 유저프로세스는 직접 인터럽트 서비스 루틴 테이블을 수정할 수 없다

2. 소프트웨어인터럽트는 현재 실행중인 프로세스에 싱크로너스하게 발생한다

3. 각 쓰레드는 자신만의 힙 메모리 영역을 갖지 않는다

4. 현대 OS는 논 프리미티브 스케쥴링 알고리즘만 이용하지 않는다

5. shortest job first SJF 스케쥴링 알고리즘은 현대 OS에서 완벽히 구현가능한 optimal한 알고리즘이 아니다

6. 멀티레벨큐 스케쥴링 알고리즘은 여러 프로세스가 동일한 프라이어티 레벨을 갖는 것을 허용한다

7. 같은 프로세스에 있는 스레드들은 같은 스택영역을 공유하지 않는다

8. 좀비프로세스는 종료된 후 다시 exec() 시스템 콜을 통해 재실행된 프로세스가 아니다

9. 프로그램드 I/O방식은 cpu가 매번 io 디바이스의 상태를 체크하면서 io연산을 처리하는 방식이다

10. 어떤 쉐어드 메모리를 통해 여러 프로세스들이 통신할 때 해당 쉐어드메모리를 할당받는 것을 제외하고는 프로세스 간 통신은 커널의 개입 없이 이루어진다

11. 한 프로세스가 사용하던 cpu를 강제로 빼앗아 다른 프로세스에게 할당해 주는 것 프림션이 일어날 수 있는 경우
    -프로세스에 할당된 타임슬라이스가 만료 되었을 때 O
    -프로세스가 io 이벤트를 기다릴 때 X
    -프로세스가 종료되었을 때 X
    -위 세가지 경우 모두에 해당될 때

12. MS DOS에 대한 설명
    -시스템이 부팅되면 쉘 프로그램이 실행된다 O
    -여러 프로세스들을 동시에 실행시킬 수 있다 X
    -유저모드와 커널모드에 대한 구분이 없다 O
    -심플 스트럭쳐의 OS구조를 사용한다 O

13. DMA io 처리 방식
    -DMA 컨트롤러가 CPU를 대신해서 io연산을 처리하는 방식이다 O
    -io연산이 수행되는 동안 cpu는 다른 일을 수행할 수 있다 O
    -빠른 속도의 io디바이스를 지원하는데 적합하다 O
    -io연산이 끝나면 DMA 컨트롤러가 cpu에게 트랩을 발생시킨다 X

14. OS 스트럭쳐
    -마이크로커널 구조는 확장하기 쉽고 시스템을 더 안정적으로 만들 수 있다 O
    -로더블 커널 모듈들 사이에서는 메시지 패싱 통신을 필요로 한다 X

-모노리틱구조는 코드 사이에 존재하는 상호의존성 때문에 코드 유지보수가 어렵다 O
-레이어드 어프로치는 OS를 기능별로 여러 개의 레이어들로 나누어 구성하는 OS 디자인 방법이다 O

15. 프로세스
-exec() 시스템 콜을 통해 새로운 프로세스를 생성할 수 있다 X
-프로세스는 오직 하나의 실행흐름만을 가질 수 있다 X
-프로세스는 커널에서 제공되는 디폴트싱글핸들러를 통해서만 시그널을 처리할 수 있다 X
-프로세스는 kill()시스템 콜을 통해 다른 프로세스에게 시그널을 보낼 수 있다 O

15-4