내용 확인문제

답 안 1.

동기 입출력이란 데이터의 송수신이 완료되는 시점이 호출된 함수가 반환을 하는 시점과 일치하는 입출력 방식을 말한다. 즉, send 함수가 반환을 하는 시점이 데이터의 전송이 완료되는 시점이고, recv 함수가 반환을 하는 시점이 데이터의 수신이 완료되는 시점이다. 그러나 비동기 입출력은 데이터의 송수신이 완료되는 시점과 상관없이 호출된 함수가 반환되는 입출력 방식이다. 동기 입출력의 단점은 입출력이 완료될 때까지 블로킹 상태에 놓인다는 점이다. 이 시간에는 사실 CPU의 연산이 거의 수반되지 않음에도 불구하고 블로킹 상태에 놓여서 다른 작업을 진행할 수 없게 된다. 이런 측면에서 볼 때 비동기 입출력은 CPU를 보다 효율적으로 사용하는 입출력 방식이라 할 수 있다.

답 안 2.

비동기 입출력의 장점이자 단점은 입출력의 완료를 이후에 확인해야 한다는 것이다. 이는 서버의 서비스 형태가 매우 간단하고, 응답에 필요한 데이터의 크기가 작은 경우에는 불편하게 느껴질 수 있다. 특히 클라이언트 하나당 쓰레드를 하나씩 생성하는 서버 모델에서는 굳이 비동기로 입출력을 진행할 필요가 없다.

답 안 3.

🞄 select 방식은 호출된 함수의 반환을 통해서 IO 관련 이벤트의 발생을 알리니, Notification IO 모델이라 할 수 있다. ( 🌕 )

🞄 select 방식은 IO 관련 이벤트의 발생시점과 호출된 함수의 반환시점이 일치하기 때문에 비동기 모델이 아니다. ( 🌕 )

🞄 WSAEventSelect 함수는 select 방식의 비동기 모델이라 할 수 있다. IO 관련 이벤트의 발생을 비동기의 형태로 알리기 때문이다. ( 🌕 )

답 안 4.

select 함수를 이용할 경우 관찰의 대상이 되는 핸들의 정보를 select 함수호출 시마다 매번 전달해야 한다. 반면에 WSAEventSelect 함수를 이용할 경우에는 관찰대상의 정보를 운영체제가 기록해 두기 때문에 매번 전달할 필요가 없다. 그리고 select 함수를 이용할 경우, 이벤트가 발생할 때까지 블로킹 상태에 있어야 하지만, WSAEventSelect 함수를 이용할 경우에는 이벤트가 발생할 때까지 블로킹 상태에 있을 필요가 없다. 일단 호출한 WSAEventSelect 함수를 빠져 나와서 이후에 별도의 과정을 거쳐서 이벤트의 발생유무를 확인하면 된다.

답 안 5.

비동기 입출력이 잘 어울리는 모드는 엣지 트리거 모드이다. 엣지 트리거 모드는 입력버퍼에 데이터가 남아있다고 해서 이벤트를 등록하지는 않는다. 즉, 엣지 트리거 모드에서는 비동기로 입출력이 진행중인 상황에 대해서는 이벤트를 등록하지 않기 때문에, 새로 등록되는 이벤트에 대해서만 신경을 쓰면 된다. 그러나 레벨 트리거 모드에서는 입력버퍼에 데이터가 남아있는 상황에서도 이를 이벤트로 등록한다. 따라서 레벨 트리거 모드에서 비동기로 입출력을 진행할 경우, 새로 등록된 입출력과 처리중인 입출력의 구분이 불가능하기 때문에 코드의 구현에 큰 어려움을 겪게 된다.

답 안 6.

epoll 방식도 이벤트의 관찰 대상을 등록하는 과정과 이벤트의 발생을 확인하는 과정이 별도의 함수로 분리되어 있다. 따라서 이벤트가 발생한 이후에 원하는 시점에서 이벤트의 발생유무를 확인할 수 있다. 그러므로 epoll 역시 비동기 입출력 모델이라 할 수 있다.

답 안 7.

상수 WSA\_MAXIMUM\_WAIT\_EVENTS의 값을 확인하면 된다.

답 안 8.

이벤트의 발생유무를 확인하기 위해서 1차로 WSAWaitForMultipleEvents 함수가 호출되는데, 이 함수의 호출 이후에 실제 이벤트의 발생대상을 찾기 위해서 2차로 WSAWaitForMultipleEvents 함수를 호출해야 한다. 따라서 manual-reset 모드가 아니라면, 자동으로 non-signaled 상태로 변경되기 때문에 2차로 WSAWaitForMultipleEvents 함수를 호출할 수 없게 된다.