답안 1.

UDP는 TCP와 달리 흐름제어를 진행하지 않는다. 이 흐름제어는 소켓의 연결과 종료, 그리고 데이터 송수신 전반에 걸쳐서 진행되기 때문에 TCP의 데이터 전송은 언제나 신뢰할 수 있다. 반면 UDP는 이러한 흐름제어를 진행하지 않으니, 데이터의 전송을 신뢰할 수 없다. 하지만 UDP는 흐름제어를 진행하지 않기 때문에 TCP보다 빠른 데이터의 송수신이 가능하다는 이점이 있다.

답안 2 b, c, e

답안 3

IP는 목적지 호스트로의 데이터 전달을 담당한다. 반면 UDP는 목적지 호스트에 전달된 데이터의 PORT번호를 확인해서 최종 목적지인 소켓으로의 데이터 전달을 담당한다.

답안 4

UDP는 TCP와 달리 두 소켓간의 연결 및 연결해제의 과정을 거치지 않는다. 때문에 매우 빈번한 연결 및 연결해제의 과정이 요구되는 경우에는 TCP 기반의 데이터 송수신보다 UDP 기반의 데이터 송수신이 좋은 성능을 보인다. 정리하면 다음과 같다.

"한번에 실제 송수신하는 데이터의 양은 적지만 매우 빈번한 데이터의 송수신이 요구되어서 TCP 기반에서 구현할 경우 빈번한 연결 및 연결해제의 과정을 거쳐야 하는 경우"

답안 5.

sendto 함수호출 시까지 주소정보가 할당되지 않았다면, sendto 함수가 처음 호출되는 시점에 해당 소켓에 IP와 PORT번호가 자동으로 할당된다. 또한 이렇게 한번 할당되면 프로그램이 종료될 때까지 주소정보가 그대로 유지되기 때문에 다른 UDP 소켓과 데이터를 주고받을 수 있다. 물론 IP는 호스트의 IP로, PORT는 사용하지 않는 PORT번호 하나를 임의로 골라서 할당하게 된다.

답안 6.

UDP 소켓을 대상으로 sendto 함수가 호출될 때마다 다음의 과정을 거친다.

🞄 1단계 UDP 소켓에 목적지의 IP와 PORT정보 등록

🞄 2단계 데이터 전송

🞄 3단계 UDP 소켓에 등록된 목적지 정보 삭제

이 중에서 데이터를 전송할 때마다 반복되는 1단계와 3단계 과정을 connect 함수를 호출하면 생략 가능하다. 물론 connect 함수를 호출한다고 해서 연결의 과정을 거치는 것은 아니며, 다만 IP와 PORT번호를 UDP 소켓에 지정하는 효과만 얻게 된다. 그리고 이렇게 connect 함수호출 후에는 sendto, recvfrom 함수가 아닌 write, read 함수의 호출로도 데이터를 송수신할 수 있다.