1. How CRC and checksum are produced in real protocols? Why CRC located at the tail of a packet? Checksum located at the header tail of a packet?

⇒ Checksum : TCP/IP 의 5계층 중 Layer 3, 4, 5에서 program으로 사용

사용 방법
1) data를 16bits로 나눠서 checksum(32bits)에 누적으로 더한다.
2) 그 32bits를 16bits로 나눠서 더한다.
3) 1의 보수를 취한다. (checksum 완성)

오류 탐지 방법 : checksum과 data를 16bits단위로 더하여 1111 1111 이 나오면 오류가 없다.

∴ checksum을 구한 이후에 data가 전송되야 함으로 data의 header에 존재

⇒ (k-bits) CRC : TCP/IP의 5계층 중 Layer 2에서 Hardware로 사용

사용 방법
1) n-bits의 data가 있을 때, 이를 k-bits 자리만큼 올린다.
2) 미리 약속된 k+1비트의 값으로 나눈다. (나머지 = r-bits, n>r) → 뺄셈은 XOR로 계산한다.
3) r-bits의 나머지 = CRC

오류 탐지 방법 : 수신된 n+r bits의 data를 k 값으로 나누어 나머지가 0이면 오류가 없다.

∴ hardware는 전송해야 할 data를 즉시 읽을 수 있으므로 대기 시간을 줄이기 위하여 data가 전송될 때 CRC를 계산하게 된다. 따라서 CRC는 data의 tail에 붙는다.

2. What can a RAC scheme achieve that a single parity bit scheme cannot?

⇒ SPC(single parity checking)는 1bit를 이용하여 Error Detection(에러 감지)을 할 수는 있으나 어떤 bit에서 에러가 났는지는 확인할 수 없다.

⇒ RAC(row and column)는 data를 행렬 형식으로 나열하여 row에 대한 parity, column에 대한 parity를 각각 구함으로써 error를 감지함과 동시에 1bit에 한해서 error를 수정할 수 있다.