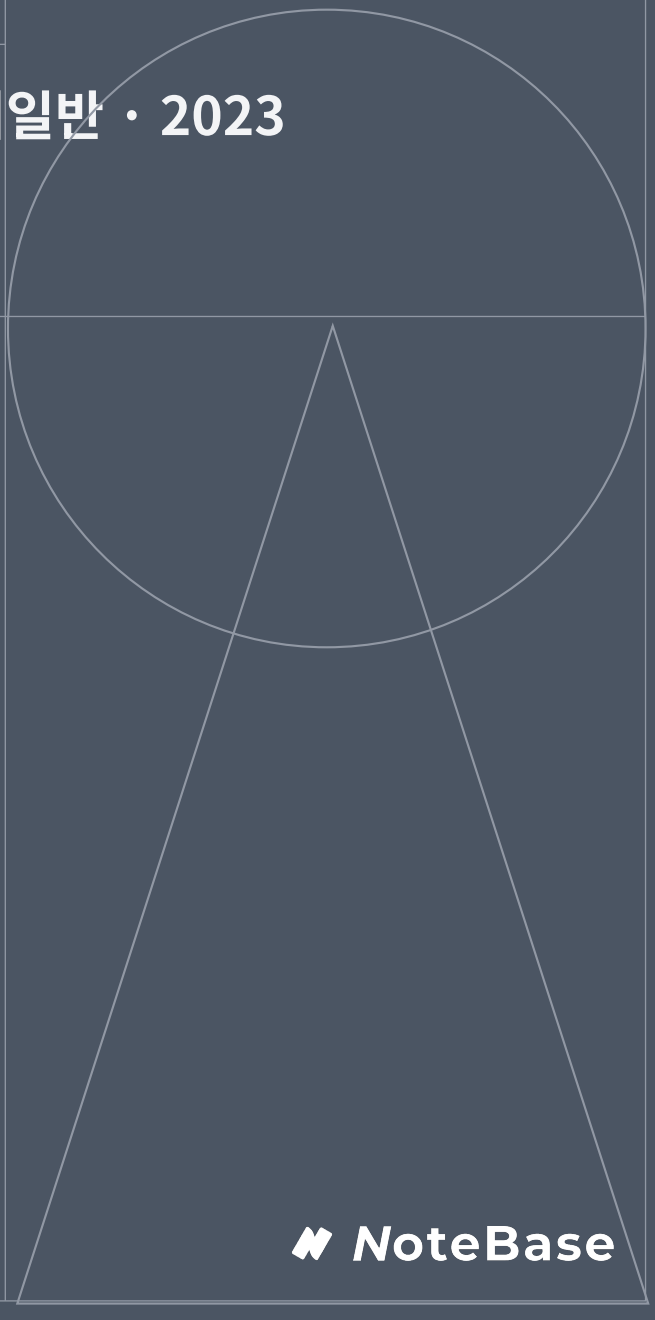
기출OX 해설집

공무원/계리직/9급 · 컴퓨터일반 · 2023



NoteBase

기출OX 해설집

공무원/계리직/9급 · 컴퓨터일반 · 2023

공무원/계리직/9급 · 컴퓨터일반 · 2023

1. 엑셀 시트를 이용한 수식의 결과값으로 옳은 것의 총 개수는?

수식	결과
=FACT(5)	15
=INT(−3.14)	−3
=MOD(3, 4)	1
=POWER(3, 3)	27
=PRODUCT(3, 6, 2)	36

① 2개 ② 3개 ③ 4개 ④ 5개

정답 1번

출제 포인트

엑셀의 FACT, INT, MOD, POWER, PRODUCT 함수가 반환하는 값을 각각 계산하여 표의 결과가 옳은 행의 개수를 구하는 문제이다.

조건 정리

주어진 값: 5개의 엑셀 수식과 제시된 결과이다.
구해야 할 값: 수식과 결과가 일치하는 행의 총개수이다.
주의할 조건: INT는 단순히 소수점 이하를 버리는 함수가 아니라 주어진 수 이하의 가장 가까운 정수로 내림하며, MOD는 나머지를 구한다.

사용할 공식

FACT(5)는 $5! = 5 \times 4 \times 3 \times 2 \times 1$을 계산한다. POWER(3, 3)은 3^3을 계산하고, PRODUCT(3, 6, 2)는 $3 \times 6 \times 2$를 계산한다. INT는 수를 내림한 정수를, MOD는 나눗셈의 나머지를 반환한다. 각 함수의 정의에 따른 실제 반환값을 표의 결과와 비교해야 하므로 이 계산 규칙을 사용한다.

NoteBase

계산 과정

FACT(5): 5! = 120이므로 제시된 15는 틀리다.
INT(-3.14): 음의 방향으로 내림하면 -4이므로 제시된 -3은 틀리다.
MOD(3, 4): 3을 4로 나눈 나머지는 3이므로 제시된 1은 틀리다.
POWER(3, 3): $3^3 = 27$이므로 제시된 27은 맞다.
PRODUCT(3, 6, 2): $3 \times 6 \times 2 = 36$이므로 제시된 36은 맞다.
따라서 옳은 행은 POWER와 PRODUCT의 2개이다.

선지 판단

① O — 실제로 옳은 행은 POWER와 PRODUCT의 2개이므로 '2개'가 맞다.
② X — '3개'는 틀린 개수이다. 올바르게는 '2개'이다.
③ X — '4개'는 틀린 개수이다. 올바르게는 '2개'이다.
④ X — '5개'라면 모든 결과가 맞아야 하지만 FACT, INT, MOD의 결과가 틀리다. 올바르게는 '2개'이다.

오답 포인트

INT(-3.14)를 소수점 이하 단순 삭제로 생각해 -3으로 판단하거나, MOD(3, 4)를 몫 또는 1로 잘못 계산하기 쉽다. INT의 음수 내림과 MOD의 나머지를 정확히 구분해야 한다.

다시 나오면 이렇게 판단

각 함수의 뜻을 먼저 확인하고 실제 반환값을 한 행씩 계산한 뒤, 제시된 결과와 일치하는 행에만 표시하여 마지막에 개수를 센다.

암기 포인트

FACT는 계승, INT는 내림 정수, MOD는 나머지, POWER는 거듭제곱, PRODUCT는 곱이다.

NoteBase

2. 자료구조가 정수형으로 이루어진 스택이며, 초기에는 빈 스택이라고 할 때, 빈칸 ㉠~㉢의 내용으로 모두 옳은 것은? (단, top()은 스택의 최상위 원소값을 출력하는 연산이다.)

연산	출력	스택 내용
push(7)	–	(7)
push(4)	–	(7, 4)
push(1)	–	(7, 4, 1)
pop()	–	㉠
㉡	–	(7)
top()	㉢	(7)
push(5)	–	(7, 5)

	㉠	㉡	㉢
①	(7, 4)	push()	1
②	(4, 1)	push(7)	1
③	(7, 4)	pop()	7
④	(4, 1)	pop(7)	7

정답 3번

출제 포인트

스택의 후입선출 원리와 push(), pop(), top()을 차례로 수행했을 때의 스택 내용 및 출력값을 묻는 문제이다. 스택에서는 가장 나중에 들어간 오른쪽 끝 원소가 최상위 원소가 된다.

올바른 순서

1. push(7), push(4), push(1)을 수행하면 스택은 (7, 4, 1)이 된다.
2. 첫 번째 pop()은 최상위 원소 1을 제거하므로 ㄱ은 (7, 4)이다.
3. 다음 스택 내용이 (7)이 되려면 최상위 원소 4를 제거해야 하므로 ㄴ은 pop()이다.
4. top()은 현재 최상위 원소 7을 출력하되 제거하지 않으므로 ㄷ은 7이고 스택은 (7)로 유지된다.
5. 마지막 push(5)를 수행하면 스택은 (7, 5)가 된다.

왜 이 순서: 각 연산의 결과가 바로 다음 연산의 입력 상태가 되므로, 연산을 건너뛰지 않고 위에서 아래로 추적해야 한다.

NoteBase

선지별 해설

① X — ㄱ의 (7, 4)는 맞지만, (7, 4)에서 (7)로 만들려면 4를 제거하는 pop()이 필요하고 top()의 출력도 1이 아니라 7이다. 올바르게는 'ㄱ (7, 4), ㄴ pop(), ㄷ 7'이다.
② X — 첫 번째 pop()은 최상위 원소 1만 제거하므로 ㄱ은 (4, 1)이 아니라 (7, 4)이다. 또한 ㄴ은 push(7)이 아니라 pop(), ㄷ은 1이 아니라 7이다. 올바르게는 'ㄱ (7, 4), ㄴ pop(), ㄷ 7'이다.
③ O — 첫 번째 pop() 후에는 (7, 4), 두 번째 pop() 후에는 (7)이 되며, 이어지는 top()은 최상위 원소 7을 출력한다. 따라서 ㄱ (7, 4), ㄴ pop(), ㄷ 7이 모두 옳다.
④ X — 첫 번째 pop() 후 ㄱ은 (4, 1)이 아니라 (7, 4)이다. pop은 삭제할 값을 인수로 받는 pop(7)이 아니라 인수 없이 pop()으로 표기하며, 이때 제거되는 값은 현재 최상위 원소 4이다. 올바르게는 'ㄱ (7, 4), ㄴ pop(), ㄷ 7'이다.

오답 포인트

스택에서 먼저 들어간 7을 먼저 제거하는 것으로 착각하거나, top()이 원소를 출력한 뒤 삭제한다고 혼동하기 쉽다. 이 문항에서는 오른쪽 끝이 최상위이며, pop()은 최상위 원소를 제거하고 top()은 출력만 하므로 스택 내용이 바뀌지 않는다.

다시 나오면 이렇게 판단

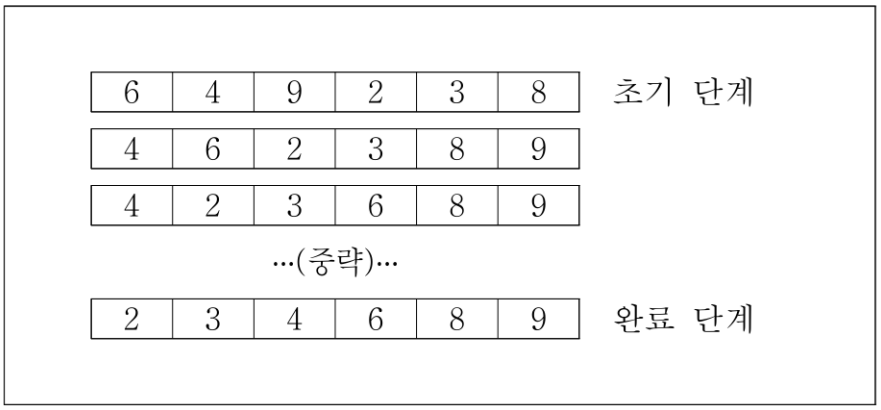
빈 스택부터 시작해 연산 한 줄마다 현재 스택을 다시 적는다. push(x)는 오른쪽 끝에 x를 추가하고, pop()은 오른쪽 끝 원소를 제거하며, top()은 오른쪽 끝 원소를 읽기만 한다는 기준으로 판단한다.

암기 포인트

스택은 후입선출이다. push는 추가, pop은 제거, top은 확인만 한다.

NoteBase

3. 다음은 정렬 알고리즘을 이용해 초기 단계의 데이터를 완료 단계의 데이터로 정렬하는 과정을 보여 준다. 이 과정에 사용된 정렬 알고리즘으로 옳은 것은?



① 퀵(quick) 정렬
② 기수(radix) 정렬
③ 버블(bubble) 정렬
④ 합병(merge) 정렬

정답 3번

출제 포인트

인접한 두 원소를 차례로 비교·교환할 때 큰 값이 매 회전마다 오른쪽 끝으로 이동하는 버블 정렬의 진행 과정을 판별하는 문제이다.

올바른 순서

1. 초기 배열 6, 4, 9, 2, 3, 8에서 왼쪽부터 인접한 두 원소를 비교하고, 앞의 값이 더 크면 서로 바꾼다.
2. 첫 번째 회전이 끝나면 가장 큰 9가 맨 오른쪽에 확정되어 4, 6, 2, 3, 8, 9가 된다.
3. 두 번째 회전이 끝나면 남은 부분의 가장 큰 8이 그 왼쪽에 확정되어 4, 2, 3, 6, 8, 9가 된다.
4. 같은 과정을 반복하면 2, 3, 4, 6, 8, 9가 되어 정렬이 완료된다.

왜 이 순서: 왼쪽에서 오른쪽으로 인접 원소를 연속 비교하면 큰 값이 한 칸씩 오른쪽으로 이동하므로, 각 회전의 결과로 정렬된 오른쪽 구간이 하나씩 늘어난다.

NoteBase

선지별 해설

① X — 퀵 정렬은 피벗을 기준으로 작은 값과 큰 값을 분할한 뒤 각 부분을 재귀적으로 정렬한다. 제시된 과정처럼 인접 원소를 차례로 교환하여 매 회전마다 최댓값을 오른쪽에 확정하는 방식이 아니다. 틀린 표현인 '퀵 정렬'은 '버블 정렬'로 고쳐야 한다.
② X — 기수 정렬은 일의 자리, 십의 자리처럼 자릿값을 기준으로 원소를 분배하고 수집한다. 그림에는 자릿수별 분배 과정이 없고 인접 원소의 교환 결과가 나타난다. 틀린 표현인 '기수 정렬'은 '버블 정렬'로 고쳐야 한다.
③ O — 첫 번째 회전에서 9가 맨 끝으로 이동하고, 다음 회전에서 8이 그 앞에 자리 잡는다. 이는 인접 원소를 비교·교환하여 큰 값부터 오른쪽에 확정하는 버블 정렬의 전형적인 진행 과정이다.
④ X — 합병 정렬은 배열을 작은 부분 배열로 계속 나눈 뒤 정렬된 부분 배열들을 합치는 방식이다. 제시된 중간 배열은 분할·합병 결과가 아니라 인접 비교·교환의 결과이다. 틀린 표현인 '합병 정렬'은 '버블 정렬'로 고쳐야 한다.

오답 포인트

정렬의 최종 결과만 보면 네 알고리즘 모두 오름차순 정렬이 가능하므로 구별할 수 없다. 첫 번째 중간 배열에서 최댓값 9가 오른쪽 끝에 확정되고, 다음 중간 배열에서 그다음 큰 값 8이 바로 앞에 확정되는 이동 규칙을 확인해야 한다.

다시 나오면 이렇게 판단

중간 배열에서 인접 원소끼리 교환되면서 한 번의 회전마다 가장 큰 값이 오른쪽 끝으로 밀려나는지 확인한다. 이 패턴이면 버블 정렬로 판단한다. 피벗 중심 분할은 퀵 정렬, 자릿수별 분배는 기수 정렬, 분할 후 정렬된 묶음의 결합은 합병 정렬이다.

암기 포인트

버블 정렬은 인접 비교·교환을 반복하며 큰 값이 거품처럼 오른쪽 끝으로 떠오른다.

NoteBase

4. <보기>에서 해시 함수(hash function)의 충돌 해결 방안으로 옳은 것의 총 개수는?

< 보 기 >

ㄱ. 별도 체이닝(separate chaining)
ㄴ. 오픈 어드레싱(open addressing)
ㄷ. 선형 검사(linear probing)
ㄹ. 이중 해싱(double hashing)

① 1개 ② 2개 ③ 3개 ④ 4개

정답 4번

출제 포인트

해시 충돌 해결 방법인 별도 체이닝과 오픈 어드레싱을 구분하고, 선형 검사와 이중 해싱이 오픈 어드레싱의 구체적인 탐사 방법임을 판단하는 문제이다.

핵심 개념

별도 체이닝: 같은 해시 주소에 여러 키가 충돌하면 해당 버킷에 연결 리스트 등의 별도 구조를 두어 함께 저장하는 방법이다.
오픈 어드레싱: 충돌이 발생하면 해시 테이블 안의 다른 빈 슬롯을 탐사하여 저장하는 방법이다.
선형 검사: 오픈 어드레싱에서 일정한 간격으로 슬롯을 차례대로 확인하는 방법이다.
이중 해싱: 오픈 어드레싱에서 두 번째 해시 함수로 탐사 간격을 정하는 방법이다.
따라서 별도 체이닝과 오픈 어드레싱은 대표적인 충돌 해결 방식이고, 선형 검사와 이중 해싱은 오픈 어드레싱에 속하는 충돌 해결 기법이다.

선지별 해설

① O — ㄱ의 별도 체이닝은 충돌한 원소를 동일 버킷의 별도 연결 구조에 저장하므로 해시 충돌 해결 방법이다.
② O — ㄴ의 오픈 어드레싱은 충돌 시 테이블 내부의 다른 빈 슬롯을 찾아 저장하므로 해시 충돌 해결 방법이다.
③ O — ㄷ의 선형 검사는 충돌한 위치부터 슬롯을 순차적으로 탐사하는 오픈 어드레싱 방식이므로 해시 충돌 해결 방법이다.
④ O — ㄹ의 이중 해싱은 두 번째 해시 함수로 탐사 간격을 정하는 오픈 어드레싱 방식이므로 해시 충돌 해결 방법이다.
옳은 항목은 ㄱ, ㄴ, ㄷ, ㄹ의 4개이므로 ④가 정답이다.

NoteBase

오답 포인트

선형 검사와 이중 해싱이 오픈 어드레싱의 하위 기법이라는 이유로 별도의 충돌 해결 방법이 아니라고 제외하기 쉽다. 그러나 각 항목은 모두 충돌을 해결하는 올바른 방식 또는 구체적 기법이므로 네 항목을 모두 센다.

다시 나오면 이렇게 판단

충돌한 데이터를 테이블 바깥의 연결 구조에 보관하면 별도 체이닝, 테이블 안의 빈 슬롯을 탐사하면 오픈 어드레싱으로 판단한다. 오픈 어드레싱의 대표 탐사법에는 선형 검사와 이중 해싱이 포함된다.

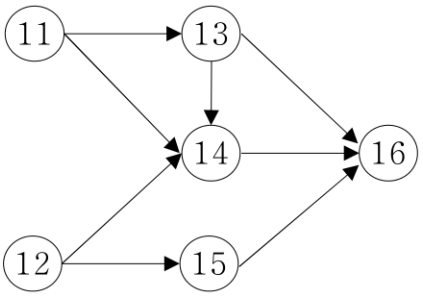
암기 포인트

충돌 해결은 체이닝 또는 빈 슬롯 탐사이며, 선형 검사와 이중 해싱은 오픈 어드레싱의 탐사법이다.

NoteBase

5. 다음은 위상 정렬의 예이다. 위상 순서로 옳은 것은?

과목 코드	과목명	선수과목
11	전산개론	없음
12	이산수학	없음
13	자바	11
14	알고리즘	11, 12, 13
15	수치해석	12
16	캡스톤디자인	13, 14, 15



① 11, 12, 14, 13, 15, 16

② 12, 11, 13, 14, 15, 16

③ 13, 11, 14, 12, 15, 16

④ 14, 13, 12, 15, 11, 16

정답 2번

출제 포인트

위상 정렬은 방향 그래프에서 모든 간선의 출발 정점이 도착 정점보다 먼저 오도록 정점을 나열하는 방법이다. 이 문제에서는 각 선수과목이 해당 과목보다 앞에 배치되어야 한다.

NoteBase

선지별 해설

① X
11, 12, 14, 13, 15, 16에서는 14가 선수과목인 13보다 먼저 나온다. 틀린 부분은 '14, 13'의 순서이며, 올바르게는 13을 14보다 앞에 두어야 한다. 예를 들면 11, 12, 13, 14, 15, 16이 가능하다.

② O
12, 11, 13, 14, 15, 16은 11→13, 11→14, 12→14, 13→14, 12→15, 13→16, 14→16, 15→16의 모든 선후관계를 만족한다. 따라서 올바른 위상 순서이다.

③ X
13, 11, 14, 12, 15, 16에서는 13이 선수과목인 11보다 먼저 나오고, 14도 선수과목인 12보다 먼저 나온다. 틀린 부분은 '13, 11' 및 12보다 앞선 14의 위치이며, 올바르게는 11을 13보다 앞에, 12를 14보다 앞에 두어야 한다. 예를 들면 12, 11, 13, 14, 15, 16이 가능하다.

④ X
14, 13, 12, 15, 11, 16에서는 14가 선수과목인 11, 12, 13보다 먼저 나오며, 13도 선수과목인 11보다 먼저 나온다. 틀린 부분은 맨 앞의 14와 11보다 앞선 13의 위치이며, 올바르게는 11과 12를 먼저 배치하고 11 뒤에 13, 11·12·13 뒤에 14를 배치해야 한다. 예를 들면 12, 11, 13, 14, 15, 16이 가능하다.

오답 포인트

선수과목이 없는 11과 12의 상대적 순서는 바뀔 수 있으므로 시작 순서만 보고 판단하면 안 된다. 각 선지에서 모든 선수과목이 후속 과목보다 앞에 있는지를 하나씩 확인해야 하며, 특히 14는 11·12·13이 모두 끝난 뒤, 16은 13·14·15가 모두 끝난 뒤에 와야 한다.

다시 나오면 이렇게 판단

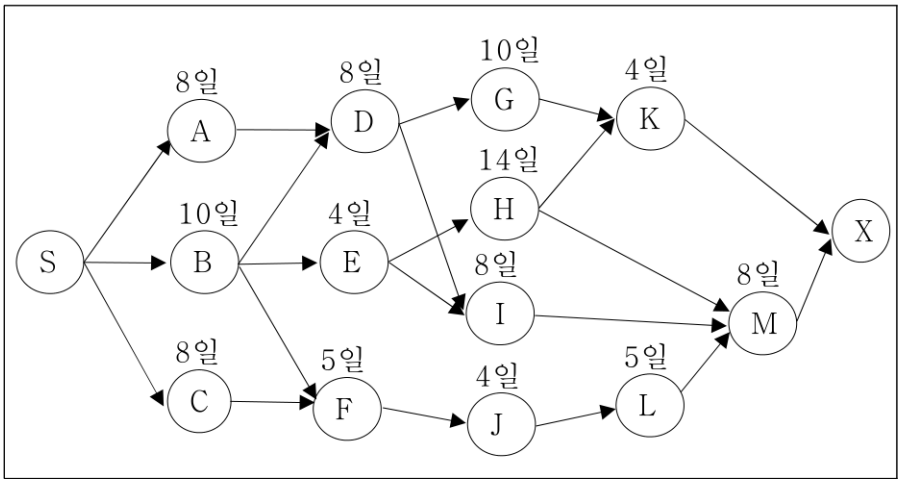
각 과목의 선수과목을 간선의 출발점으로 보고 '출발점이 도착점보다 앞에 있는가'를 모든 간선에 대해 검사한다. 간선 하나라도 순서가 뒤집히면 그 배열은 위상 순서가 아니다.

암기 포인트

위상 정렬의 핵심은 모든 선수과목이 후속 과목보다 먼저 오는 것이다.

NoteBase

6. 다음 CPM(Critical Path Method) 네트워크에 나타난 임계 경로(critical path)의 전체 소요 기간으로 옳은 것은?



① 30일 ② 32일 ③ 34일 ④ 36일

정답 4번

출제 포인트

이 문제는 CPM 네트워크에서 각 작업의 선행관계를 반영해 최조 완료시각을 계산하고, 전체 공기를 결정하는 임계 경로의 소요 기간을 구하는 문제이다.

조건 정리

주어진 값: 각 작업 노드 위의 숫자는 해당 작업의 소요 기간이다.
구해야 할 값: 시작점 S에서 종료점 X까지 프로젝트 전체를 완료하는 데 필요한 최소 기간이다.
주의할 조건: 여러 선행 작업이 한 노드로 합류하면 그중 가장 늦게 끝나는 작업이 완료되어야 다음 작업을 시작할 수 있다.

사용할 공식

작업 v의 최조 완료시각을 $EF(v)$, 작업 기간을 $d(v)$라 하면 다음과 같다.

$$EF(v) = d(v) + \max_{u \in Pred(v)} EF(u)$$

시작점은 $EF(S) = 0$으로 둔다. CPM에서는 합류점의 모든 선행 작업이 끝나야 후속 작업을 시작할 수 있으므로 선행 작업들의 완료시각 중 최댓값을 사용한다.

NoteBase

계산 과정

1. 첫 단계의 완료시각은 A=8일, B=10일, C=8일이다.
2. D=max(8, 10) + 8 = 18일, E=10 + 4 = 14일, F=max(10, 8) + 5 = 15일이다.
3. G=18 + 10 = 28일, H=14 + 14 = 28일, I=max(18, 14) + 8 = 26일, J=15 + 4 = 19일이다.
4. K=max(28, 28) + 4 = 32일, L=19 + 5 = 24일이다.
5. M=max(28, 26, 24) + 8 = 36일이다.
6. 종료점 X에는 K와 M이 연결되므로 전체 완료시각은 max(32, 36) = 36일이다. 이 값을 결정하는 임계 경로는 S-B-E-H-M-X이며, 소요 기간은 10 + 4 + 14 + 8 = 36일이다.

선지 판단

① X
30일이라는 값은 전체 완료시각 36일보다 짧다. H가 28일에 완료된 뒤 M에 8일이 더 필요하므로 M 경로만으로도 36일이 걸린다. 틀린 표현 '30일'은 '36일'로 고쳐야 한다.

② X
32일은 K의 최조 완료시각일 뿐이다. 종료점 X는 K와 M이 모두 완료되어야 도달하며 M은 36일에 완료된다. 틀린 표현 '32일'은 '36일'로 고쳐야 한다.

③ X
34일은 전진 계산에서 얻어지는 프로젝트 완료시각이 아니다. S-B-E-H-M-X의 기간을 모두 합하면 36일이다. 틀린 표현 '34일'은 '36일'로 고쳐야 한다.

④ O
임계 경로 S-B-E-H-M-X의 총기간은 10+4+14+8=36일이고, 다른 최종 경로인 K 쪽 완료시각 32일보다 길다. 따라서 전체 소요 기간은 36일이다.

오답 포인트

K가 32일에 끝난다는 이유만으로 전체 공기를 32일로 판단하면 안 된다. 종료점으로 들어오는 K와 M 중 더 늦게 완료되는 M의 36일이 전체 공기를 결정한다. 또한 합류점에서는 선행 작업 기간을 모두 더하는 것이 아니라 각 선행 경로의 완료시각 중 최댓값을 선택한다.

다시 나오면 이렇게 판단

시작점부터 각 노드의 최조 완료시각을 전진 계산한다. 합류점에서는 선행 완료시각의 최댓값에 해당 작업 기간을 더하고, 종료점의 완료시각을 구한 뒤 그 값을 만든 선행 노드를 거꾸로 추적하면 임계 경로와 전체 소요 기간을 함께 확인할 수 있다.

NoteBase

암기 포인트

CPM 전진 계산의 핵심은 ‘합류점에서는 최댓값, 마지막 값이 전체 공기’이다.

NoteBase

7. <보기>에서 디자인 패턴에 대한 설명으로 옳은 것의 총 개수는?

< 보 기 >

ㄱ. 디자인 패턴은 유사한 문제를 해결하기 위하여 각 문제 유형별로 적합한 설계를 일반화하여 정리해 놓은 것이다.

ㄴ. 싱글턴(singleton) 패턴은 특정 클래스의 객체가 오직 하나만 존재하도록 보장하여 객체가 불필요하게 여러 개 만들어질 필요가 없는 경우에 주로 사용한다.

ㄷ. 메멘토(memento) 패턴은 한 객체의 상태가 변경되었을 때 의존 관계에 있는 다른 객체들에게 이를 자동으로 통지하도록 하는 패턴이다.

ㄹ. 데코레이터(decorator) 패턴은 기존에 구현된 클래스의 기능 확장을 위하여 상속을 활용하는 설계 방안을 제공한다.

① 1개 ② 2개 ③ 3개 ④ 4개

정답 2번

출제 포인트

디자인 패턴의 일반적 의미와 싱글턴·메멘토·데코레이터 패턴의 핵심 목적을 구별하여 옳은 설명의 개수를 판단하는 문제이다.

핵심 개념

디자인 패턴: 반복해서 나타나는 설계 문제에 적용할 수 있도록 검증된 해결 구조를 일반화한 것이다.
싱글턴 패턴: 특정 클래스의 인스턴스가 하나만 생성되도록 보장하고 그 인스턴스에 접근하게 하는 패턴이다.
메멘토 패턴: 객체의 캡슐화를 해치지 않으면서 이전 상태를 저장해 두었다가 복원할 수 있게 하는 패턴이다.
데코레이터 패턴: 객체를 감싸는 합성 방식으로 기능이나 책임을 동적으로 추가하는 패턴이다.

NoteBase

선지별 해설

① X — 옳은 설명은 ㄱ과 ㄴ의 2개이므로 '1개'가 틀렸다. 올바르게는 '2개'이다.
② O — ㄱ은 디자인 패턴의 정의에 부합하고, ㄴ은 싱글턴 패턴의 목적을 정확히 설명한다. 따라서 옳은 설명은 2개이다.
③ X — 옳은 설명은 2개이므로 '3개'가 틀렸다. 올바르게는 '2개'이다. ㄷ의 자동 통지는 메멘토가 아니라 옵서버 패턴의 설명이다.
④ X — 옳은 설명은 2개이므로 '4개'가 틀렸다. 올바르게는 '2개'이다. ㄹ의 데코레이터 패턴은 상속을 활용하는 방식이 아니라 객체 합성을 이용해 기능을 동적으로 확장한다.

오답 포인트

ㄷ의 '상태 변경을 다른 객체에 자동 통지'는 메멘토가 아니라 옵서버 패턴의 핵심이다. 또한 ㄹ은 데코레이터가 기능을 확장한다는 결과만 맞고, 그 수단을 '상속'이라고 한 부분이 틀렸다. 데코레이터의 핵심 수단은 객체 합성이다.

다시 나오면 이렇게 판단

패턴 설명에서 목적과 구현 수단을 함께 확인한다. 인스턴스 하나는 싱글턴, 상태 저장·복원은 메멘토, 상태 변화 통지는 옵서버, 객체를 감싸 기능을 추가하는 것은 데코레이터로 구별한 뒤 옳은 보기의 개수를 센다.

암기 포인트

하나는 싱글턴, 되돌리기는 메멘토, 알림은 옵서버, 감싸서 확장은 데코레이터이다.

NoteBase

8. 다음과 같이 '인사'로 시작하는 모든 부서에 속한 직원들의 봉급을 10% 올리고자 SQL문을 작성하였다. ㉠과 ㉡의 내용으로 옳은 것은?

```
UPDATE 직원
SET 봉급 = 봉급*1.1
WHERE 부서번호 ___㉠___ ( SELECT 부서번호
                          FROM 부서
                          WHERE 부서명 ___㉡___ '인사%' )
```

	㉠	㉡
①	IN	LIKE
②	EXISTS	HAVING
③	AMONG	LIKE
④	AS	HAVING

정답 1번

출제 포인트

서브쿼리가 반환한 여러 값과 비교하는 IN 연산자와 문자열 패턴을 검색하는 LIKE 연산자의 쓰임을 묻는 문제이다.

핵심 개념

IN: 왼쪽 값이 목록이나 서브쿼리 결과 중 하나와 일치하는지를 검사하는 연산자이다.
LIKE: 문자열이 지정한 패턴과 일치하는지를 검사하는 연산자이며, %는 길이가 0 이상인 임의의 문자열을 뜻한다. 따라서 부서명 LIKE '인사%'는 부서명이 '인사'로 시작한다는 뜻이다.
서브쿼리는 조건에 맞는 부서번호들을 반환하므로 직원의 부서번호는 그 결과에 IN으로 포함되는지 비교해야 한다.

NoteBase

선지별 해설

① O — ㉠에 IN을 쓰면 직원의 부서번호가 서브쿼리에서 반환한 부서번호 중 하나인지 검사할 수 있고, ㉡에 LIKE를 쓰면 부서명이 '인사'로 시작하는 부서를 찾을 수 있다. 이 직원들의 봉급에 1.1을 곱하므로 봉급이 10% 인상된다.
② X — EXISTS는 서브쿼리 결과 행의 존재 여부를 검사하므로 '부서번호 EXISTS (서브쿼리)'와 같이 값의 소속 여부를 비교할 수 없다. HAVING은 주로 GROUP BY로 만든 그룹을 거르는 절이지 부서명의 문자열 패턴을 비교하는 연산자가 아니다. 올바른 표현은 ㉠ IN, ㉡ LIKE이다.
③ X — AMONG은 이 위치에서 값이 서브쿼리 결과에 포함되는지를 검사하는 SQL 연산자가 아니다. ㉡의 LIKE는 맞지만, ㉠은 IN으로 고쳐야 한다. 올바른 표현은 ㉠ IN, ㉡ LIKE이다.
④ X — AS는 열이나 테이블에 별칭을 지정할 때 쓰며 값의 포함 여부를 검사하지 않는다. HAVING도 부서명의 접두 문자열을 검색하는 연산자가 아니다. 올바른 표현은 ㉠ IN, ㉡ LIKE이다.

오답 포인트

②의 EXISTS는 서브쿼리와 함께 쓰인다는 점 때문에 IN과 혼동하기 쉽다. 그러나 이 문장에서는 왼쪽의 부서번호를 서브쿼리가 반환한 여러 부서번호와 비교하므로 IN이 필요하다. 또한 '인사로 시작'은 그룹 조건이 아니라 문자열 패턴 조건이므로 HAVING이 아니라 LIKE를 사용한다.

다시 나오면 이렇게 판단

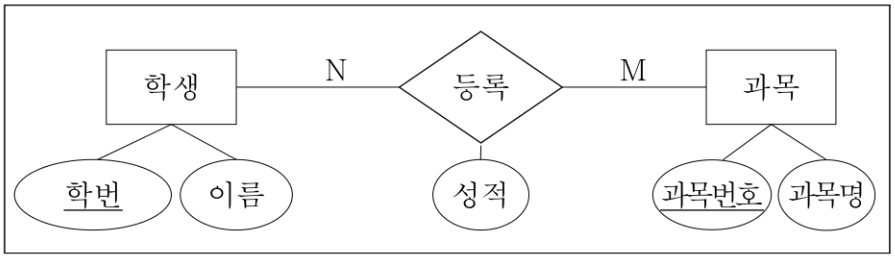
왼쪽 값이 서브쿼리 결과 목록 중 하나인지 묻는 구조이면 IN을 선택한다. 문자열의 시작·끝·포함 패턴을 찾고 %나 _ 같은 와일드카드를 사용하면 LIKE를 선택한다.

암기 포인트

여러 결과 중 하나는 IN, 문자열 패턴은 LIKE이다. '인사%'는 '인사'로 시작하는 모든 문자열이다.

NoteBase

9. 다음 E-R다이어그램을 관계형 스키마로 올바르게 변환한 것은?
(단, 속성명의 밑줄은 해당 속성이 기본키임을 의미한다.)



① 학생(학번, 이름)
등록(성적)
과목(과목번호, 과목명)

② 학생(학번, 이름)
등록(과목번호, 성적)
과목(과목번호, 과목명, 성적)

③ 학생(학번, 이름)
등록(학번, 성적)
과목(과목번호, 과목명)

④ 학생(학번, 이름)
등록(학번, 과목번호, 성적)
과목(과목번호, 과목명)

정답 4번

출제 포인트

이 문제는 학생과 과목 사이의 M:N 관계인 등록을 릴레이션으로 변환하는 방법을 묻는다. M:N 관계는 양쪽 개체의 기본키를 외래키로 가져와 함께 기본키로 삼고, 관계의 속성도 그 관계 릴레이션에 포함한다.

NoteBase

선지별 해설

① X
등록 릴레이션에 성적만 있어 학생과 과목의 어떤 조합에 대한 등록인지 식별할 수 없다. 틀린 표현 '등록(성적)'은 '등록(학번, 과목번호, 성적)'으로 고치고, 학번과 과목번호를 복합 기본키로 지정해야 한다.

② X
등록 릴레이션에 학번이 없으므로 어떤 학생의 등록인지 식별할 수 없고, 관계의 속성인 성적을 과목 릴레이션에 넣은 것도 잘못이다. 틀린 표현 '등록(과목번호, 성적), 과목(과목번호, 과목명, 성적)'은 '등록(학번, 과목번호, 성적), 과목(과목번호, 과목명)'으로 고치며, 등록의 학번과 과목번호를 복합 기본키로 지정해야 한다.

③ X
등록 릴레이션에 과목번호가 없으므로 학생이 어느 과목에 등록했는지 식별할 수 없다. 틀린 표현 '등록(학번, 성적)'은 '등록(학번, 과목번호, 성적)'으로 고치고, 학번과 과목번호를 복합 기본키로 지정해야 한다.

④ O
학생은 학생(학번, 이름), 과목은 과목(과목번호, 과목명)으로 변환한다. M:N 관계인 등록은 양쪽 기본키인 학번과 과목번호를 함께 복합 기본키로 사용하고 관계 속성인 성적을 포함하므로 올바르다.

오답 포인트

등록 관계의 속성인 성적만 옮기거나, 양쪽 기본키 중 하나만 등록 릴레이션에 넣는 것이 함정이다. 성적은 특정 학생과 특정 과목의 등록 조합에 종속되므로 학생이나 과목 릴레이션에 단독으로 둘 수 없다.

다시 나오면 이렇게 판단

E-R 다이어그램의 M:N 관계를 찾은 뒤 별도의 관계 릴레이션을 만든다. 그 릴레이션에 두 개체의 기본키를 모두 포함하여 복합 기본키로 삼고, 관계에 연결된 속성을 함께 넣었는지 확인한다.

암기 포인트

M:N 관계는 '양쪽 기본키를 합친 복합 기본키 + 관계 속성'으로 변환한다.

NoteBase

10. OpenAI가 개발한 생성형 인공지능 기반의 대화형 서비스는?

① LSTM

② ResNET

③ ChatGPT

④ Deep Fake

정답 3번

출제 포인트

OpenAI가 개발한 생성형 인공지능 기반의 대화형 서비스와 인공지능 모델·기술을 구별하는 문제이다.

핵심 개념

ChatGPT는 OpenAI가 개발한 생성형 인공지능 기반 대화형 서비스로, 사용자의 질문이나 요청에 대화 형식으로 응답한다. LSTM은 순환 신경망 계열의 구조이고, ResNet은 잔차 연결을 사용하는 심층 신경망 구조이며, 딥페이크는 인공지능으로 영상·음성 등을 합성하거나 변조하는 기술이다.

선지별 해설

① X — LSTM은 긴 문맥의 의존 관계를 학습하도록 설계된 순환 신경망 계열의 구조이지, OpenAI가 개발한 대화형 서비스의 명칭이 아니다. 틀린 표현은 'LSTM'이며, 올바르게는 'ChatGPT'이다.
② X — ResNET(일반적으로 ResNet)은 잔차 연결을 이용해 깊은 신경망을 학습하는 모델 구조이지, 대화형 서비스가 아니다. 틀린 표현은 'ResNET'이며, 올바르게는 'ChatGPT'이다.
③ O — ChatGPT는 OpenAI가 개발한 생성형 인공지능 기반의 대화형 서비스이므로 문제의 설명에 해당한다.
④ X — Deep Fake(딥페이크)는 인공지능을 활용해 얼굴·영상·음성 등을 합성하거나 변조하는 기술을 가리키며, OpenAI의 대화형 서비스 명칭이 아니다. 틀린 표현은 'Deep Fake'이며, 올바르게는 'ChatGPT'이다.

오답 포인트

선지들이 모두 인공지능과 관련되어 있다는 점이 함정이다. LSTM과 ResNet은 신경망 구조, 딥페이크는 합성·변조 기술이고, 대화형 서비스의 고유 명칭은 ChatGPT이다.

다시 나오면 이렇게 판단

문장에서 '누가 개발했는가'와 '모델 구조·기술·서비스 중 무엇을 묻는가'를 먼저 구분한다. 'OpenAI가 개발한 생성형 인공지능 기반 대화형 서비스'라는 단서가 함께 나오면 ChatGPT로 판단한다.

암기 포인트

ChatGPT는 대화형 서비스, LSTM·ResNet은 신경망 구조, 딥페이크는 합성·변조 기술이다.

NoteBase

11. CPU 스케줄링 기법에 대한 설명으로 옳지 않은 것은?

① 라운드 로빈(Round-Robin) 스케줄링 기법은 선점 방식의 스케줄링 기법이다.

② HRN(Highest Response ratio Next) 스케줄링 기법은 우선순위에 대기 시간(waiting time)을 고려하여 기아(starvation) 문제를 해결한다.

③ 다단계 큐 스케줄링 기법은 프로세스들을 위한 준비 큐를 다수 개로 구분하며, 각 준비 큐는 자신만의 스케줄링 알고리즘을 별도로 가질 수 있다.

④ 우선순위 스케줄링 기법은 항상 선점 방식으로 구현되기 때문에 특정 프로세스에 대하여 무한대기 또는 기아(starvation) 현상 발생의 위험이 있다.

정답 4번

출제 포인트

이 문제는 라운드 로빈, HRN, 다단계 큐, 우선순위 스케줄링의 핵심 특징과 선점 가능 여부를 구별하는 문제이다.

핵심 개념

라운드 로빈은 각 프로세스에 일정한 시간 할당량을 차례로 부여하는 선점형 기법이다. HRN은 대기 시간과 서비스 시간을 함께 반영한 응답률로 다음 작업을 정한다. 다단계 큐는 프로세스 유형별로 준비 큐를 나누고 큐마다 서로 다른 스케줄링 알고리즘을 적용할 수 있다. 우선순위 스케줄링은 우선순위가 높은 프로세스를 먼저 선택하며 선점형과 비선점형으로 모두 구현할 수 있다.

선지별 해설

① O — 라운드 로빈은 실행 중인 프로세스의 시간 할당량이 끝나면 CPU를 회수하여 다음 프로세스에 넘기므로 선점형 스케줄링 기법이다.

② O — HRN은 대기 시간이 길어질수록 응답률이 높아지게 하여 오래 기다린 작업의 우선순위를 높이므로 기아 문제를 방지하는 데 유리하다.

③ O — 다단계 큐는 프로세스를 여러 준비 큐로 분류하며, 각 큐에 라운드 로빈이나 선입선처리 등 별도의 스케줄링 알고리즘을 적용할 수 있다.

④ X — 틀린 표현은 '항상 선점 방식으로 구현'이다. 올바르게는 우선순위 스케줄링은 선점형 또는 비선점형으로 구현할 수 있으며, 낮은 우선순위의 프로세스에는 기아 현상이 발생할 수 있다.

NoteBase

오답 포인트

④는 우선순위가 높은 프로세스를 먼저 실행한다는 특징을 '항상 현재 프로세스를 중단시킨다'는 뜻으로 바꾼 것이 함정이다. 높은 우선순위의 새 프로세스가 도착했을 때 즉시 CPU를 빼앗으면 선점형이고, 현재 작업이 끝날 때까지 기다리면 비선점형이다.

다시 나오면 이렇게 판단

스케줄링 설명에서 '항상'이라는 표현을 먼저 점검한다. CPU를 강제로 회수하는 시간 할당량이 있는 라운드 로빈은 선점형이고, 우선순위 스케줄링은 새 프로세스가 현재 실행을 중단시키는지에 따라 선점형과 비선점형으로 나뉜다.

암기 포인트

라운드 로빈은 선점형, HRN은 대기 시간 반영, 다단계 큐는 큐별 알고리즘, 우선순위는 선점형·비선점형 모두 가능이다.

NoteBase

12. 교착상태(deadlock)와 은행원 알고리즘(banker's algorithm)에 대한 설명으로 옳은 것은?
① 교착상태는 불안전한 상태(unsafe state)에 속한다.
② 은행원 알고리즘은 교착상태 회복(recovery) 알고리즘이다.
③ 불안전한 상태(unsafe state)는 항상 교착상태로 빠지게 된다.
④ 은행원 알고리즘은 불안전한 상태(unsafe state)에서 교착상태로 전이되는 것을 거부한다.

정답 1번

출제 포인트

교착상태와 불안전한 상태의 포함 관계, 그리고 은행원 알고리즘이 교착상태 회복이 아니라 회피에 사용된다는 점을 구별하는 문제이다.

핵심 개념

안전 상태: 모든 프로세스가 필요한 자원을 받아 종료할 수 있는 안전 순서가 존재하는 상태이다.
불안전한 상태: 안전 순서를 보장할 수 없어 교착상태가 발생할 가능성이 있는 상태이지만, 반드시 이미 교착상태인 것은 아니다.
교착상태: 여러 프로세스가 서로 상대방이 보유한 자원을 기다리며 더 이상 진행하지 못하는 상태이다. 교착상태는 불안전한 상태에 포함된다.
은행원 알고리즘: 자원 할당 후에도 안전 상태가 유지되는지를 검사하여, 불안전한 상태로 만드는 자원 할당을 거부하는 교착상태 회피 알고리즘이다.

선지별 해설

① O — 교착상태에서는 모든 프로세스를 완료시킬 안전 순서가 존재하지 않으므로 불안전한 상태에 속한다.
② X — 틀린 표현은 '교착상태 회복 알고리즘'이다. 은행원 알고리즘은 교착상태가 발생한 뒤 복구하는 알고리즘이 아니라, 안전 상태를 유지하도록 자원 할당을 통제하는 교착상태 회피 알고리즘이다.
③ X — 틀린 표현은 '항상 교착상태로 빠지게 된다'이다. 불안전한 상태는 교착상태의 가능성이 있는 상태일 뿐이므로, 이후의 자원 요청과 실행 상황에 따라 교착상태가 발생하지 않을 수도 있다. 올바르게는 '불안전한 상태는 교착상태로 이어질 가능성이 있다'이다.
④ X — 틀린 표현은 '불안전한 상태에서 교착상태로 전이되는 것을 거부한다'이다. 은행원 알고리즘은 현재 상태가 안전한지 검사하고, 자원 할당 결과 안전 상태에서 불안전한 상태로 바뀌는 할당을 거부한다.

NoteBase

오답 포인트

불안전한 상태를 교착상태와 같은 뜻으로 보는 것이 핵심 함정이다. 불안전한 상태는 교착상태가 발생할 수 있는 더 넓은 범위이며, 은행원 알고리즘의 차단 시점도 '불안전 상태에서 교착상태로 갈 때'가 아니라 '안전 상태를 유지할 수 없게 될 때'이다.

다시 나오면 이렇게 판단

먼저 '안전 순서가 존재하는가'를 기준으로 상태를 구분한다. 안전 순서가 있으면 안전 상태이고, 보장되지 않으면 불안전한 상태이며, 실제로 프로세스들이 서로 기다려 진행할 수 없으면 교착상태이다. 은행원 알고리즘은 할당 후의 안전성을 미리 검사하므로 회복이 아니라 회피에 해당한다.

암기 포인트

교착상태 ⊂ 불안전 상태, 은행원 알고리즘 = 안전성 검사에 의한 교착상태 회피이다.

NoteBase

13. 하드웨어적으로 인터럽트를 요구한 장치를 찾는 기법으로, 인터럽트 선을 공유하면서 인터럽트를 발생시키는 모든 장치를 직렬로 연결하여 연결 순서에 따라 우선순위가 결정되는 방식으로 옳은 것은?

① 소프트웨어 폴링(polling) 방식
② 데이지 체인(daisy chain) 방식
③ 인터럽트 벡터(interrupt vector) 방식
④ 다수 인터럽트 선(multiple interrupt lines) 방식

정답 2번

출제 포인트

인터럽트 요청 장치를 하드웨어적으로 판별하는 방식 중, 공통 인터럽트 선과 직렬 연결을 사용하여 연결 순서대로 우선순위를 정하는 방식을 묻는 문제이다.

핵심 개념

데이지 체인 방식은 여러 장치를 직렬로 연결하고 인터럽트 확인 신호를 차례로 전달하여 요청 장치를 찾는 하드웨어 우선순위 방식이다. CPU에 가까워 확인 신호를 먼저 받는 장치일수록 우선순위가 높다.
소프트웨어 폴링은 프로그램이 정해진 순서로 각 장치의 상태를 검사하는 방식이고, 인터럽트 벡터 방식은 장치가 서비스 루틴을 식별할 수 있는 벡터 정보를 제공하는 방식이며, 다중 인터럽트 선 방식은 여러 인터럽트 선을 병렬로 두는 방식이다.

선지별 해설

① X — '소프트웨어 폴링 방식'은 틀린 표현이다. 소프트웨어 폴링은 CPU가 프로그램으로 각 장치의 상태를 정해진 순서에 따라 검사한다. 올바르게는 공통 선에 장치를 직렬로 연결하고 연결 순서로 하드웨어 우선순위를 정하는 방식은 데이지 체인 방식이다.
② O — 데이지 체인 방식은 인터럽트 확인 신호가 직렬로 연결된 장치들을 차례로 통과하며, 연결 순서에 따라 우선순위가 결정되므로 문제의 설명과 일치한다.
③ X — '인터럽트 벡터 방식'은 틀린 표현이다. 인터럽트 벡터 방식은 인터럽트를 요청한 장치가 해당 서비스 루틴을 찾을 수 있는 벡터 정보를 제공하는 방식이다. 올바르게는 장치의 직렬 연결 순서로 우선순위를 정하는 방식은 데이지 체인 방식이다.
④ X — '다수 인터럽트 선 방식'은 틀린 표현이다. 다수 인터럽트 선 방식은 여러 인터럽트 선을 두어 장치 또는 우선순위별 요청을 구분하는 병렬적 방식이다. 올바르게는 하나의 인터럽트 선을 공유하는 장치들을 직렬로 연결하는 방식은 데이지 체인 방식이다.

NoteBase

오답 포인트

‘정해진 순서로 장치를 확인한다’는 점만 보고 ① 소프트웨어 폴링과 혼동하기 쉽다. 그러나 문제의 핵심 표현은 ‘하드웨어적으로’, ‘모든 장치를 직렬로 연결’, ‘연결 순서에 따라 우선순위 결정’이며, 이 세 조건은 데이지 체인을 가리킨다.

다시 나오면 이렇게 판단

직렬 연결과 물리적 연결 순서가 보이면 데이지 체인, 프로그램이 장치 상태를 차례로 검사하면 소프트웨어 폴링, 장치가 서비스 루틴 식별 정보를 주면 인터럽트 벡터, 여러 요청선을 병렬로 사용하면 다수 인터럽트 선 방식으로 판단한다.

암기 포인트

데이지 체인은 직렬 연결, 우선순위는 연결 순서이다.

NoteBase

14. 음수와 양수를 동시에 표현하는 2진수의 표현 방법에는 부호-크기(sign-magnitude) 방식, 1의 보수 방식, 2의 보수 방식이 있다. 다음은 10진수의 양수와 음수를 3비트의 2진수로 나타낸 표이다. ㉠~㉢에 들어갈 방식을 순서대로 나열한 것은?

10진 정수	㉠	㉡	㉢
3	011	011	011
2	010	010	010
1	001	001	001
0	000	000	000
-0	100	111	-
-1	101	110	111
-2	110	101	110
-3	111	100	101
-4	-	-	100

㉠	㉡	㉢
① 1의 보수	2의 보수	부호-크기
② 2의 보수	1의 보수	부호-크기
③ 부호-크기	1의 보수	2의 보수
④ 부호-크기	2의 보수	1의 보수

정답 3번

출제 포인트

3비트로 나타낸 음수의 비트 패턴과 0의 표현 범위를 이용하여 부호-크기, 1의 보수, 2의 보수 방식을 구별하는 문제이다.

NoteBase

핵심 개념

부호-크기 방식은 최상위 비트를 부호로, 나머지 비트를 크기로 사용한다. 따라서 3은 011, -3은 111이고, +0은 000, -0은 100으로 0이 두 개이다.
1의 보수 방식은 양수의 모든 비트를 반전하여 음수를 만든다. 따라서 011의 반전인 100이 -3이고, 000과 111이 각각 +0과 -0이다.
2의 보수 방식은 1의 보수에 1을 더하여 음수를 만든다. 0은 000 하나뿐이고, 3비트에서는 -4까지 표현하므로 -1은 111, -2는 110, -3은 101, -4는 100이다.
표의 ㄱ은 부호 비트 뒤의 크기가 그대로 유지되고, ㄴ은 양수 비트가 모두 반전되며, ㄷ은 -0 없이 -4까지 표현된다. 따라서 ㄱ은 부호-크기, ㄴ은 1의 보수, ㄷ은 2의 보수이다.

선지별 해설

① X
ㄱ을 '1의 보수', ㄴ을 '2의 보수', ㄷ을 '부호-크기'로 대응한 부분이 틀리다. ㄱ의 -3은 111로 부호 비트만 1이고 크기 11이 유지되므로 부호-크기 방식이다. 올바른 대응은 ㄱ=부호-크기, ㄴ=1의 보수, ㄷ=2의 보수이다.

② X
ㄴ=1의 보수는 맞지만, ㄱ을 '2의 보수', ㄷ을 '부호-크기'로 대응한 부분이 틀리다. ㄱ에는 -0이 있고 -4가 없으므로 부호-크기이며, ㄷ에는 -0이 없고 -4가 있으므로 2의 보수이다. 올바른 대응은 ㄱ=부호-크기, ㄴ=1의 보수, ㄷ=2의 보수이다.

③ O
ㄱ은 -0이 100이고 음수의 하위 두 비트가 크기를 나타내므로 부호-크기 방식이다. ㄴ은 000의 반전인 111이 -0이고 011의 반전인 100이 -3이므로 1의 보수 방식이다. ㄷ은 -0이 없으며 100으로 -4까지 나타내므로 2의 보수 방식이다.

④ X
ㄱ=부호-크기는 맞지만, ㄴ을 '2의 보수', ㄷ을 '1의 보수'로 서로 바꾼 부분이 틀리다. -0이 111로 존재하는 ㄴ이 1의 보수이고, -0 없이 -4까지 표현하는 ㄷ이 2의 보수이다. 올바른 대응은 ㄱ=부호-크기, ㄴ=1의 보수, ㄷ=2의 보수이다.

오답 포인트

양수 0~3의 표현은 세 방식이 모두 같으므로 양수 부분만 보아서는 구별할 수 없다. 특히 ㄴ과 ㄷ은 음수 비트가 비슷하므로 -0의 존재와 -4의 표현 가능 여부를 확인해야 한다.

NoteBase

다시 나오면 이렇게 판단

먼저 -0이 있는지 확인한다. -0이 100이면 부호-크기, -0이 111이면 1의 보수이다. -0이 없고 3비트 최솟값 -4를 100으로 표현하면 2의 보수이다. 음수 하나를 양수와 대조할 때는 비트 전체 반전이면 1의 보수, 전체 반전 후 1을 더한 값이면 2의 보수이다.

암기 포인트

부호-크기와 1의 보수는 +0·-0 두 개, 2의 보수는 0 하나와 음수 한 개를 더 표현한다.

NoteBase

15. IP주소가 117.17.23.253/27인 호스트에 대한 설명으로 옳은 것은?

① 이 주소의 네트워크 주소는 117.17.23.0이다.

② 이 주소의 서브넷 마스크는 255.255.255.224이다.

③ 이 주소는 클래스 기반의 주소지정으로 C클래스 주소이다.

④ 이 주소가 포함된 네트워크에서 사용될 수 있는 IP주소는 254개이다.

정답 2번

출제 포인트

이 문제는 CIDR 표기 /27을 서브넷 마스크로 바꾸고, 블록 크기를 이용해 네트워크 주소와 사용 가능한 호스트 주소 수를 판단하는 문제이다.

조건 정리

주어진 IP 주소: 117.17.23.253/27
구해야 할 값: 서브넷 마스크, 네트워크 주소, 클래스 기반 주소 등급, 사용 가능한 호스트 주소 수
주의할 조건: /27은 앞의 27비트가 네트워크 부분이고 나머지 5비트가 호스트 부분이라는 뜻이다.

사용할 공식

호스트 비트 수는 32 − 27 = 5비트이다.
전체 주소 수는 $2^5 = 32$개이고, 일반적인 IPv4 서브넷에서 사용 가능한 호스트 주소 수는 네트워크 주소와 브로드캐스트 주소를 제외한 $2^5 - 2 = 30$개이다.
마지막 옥텟의 블록 크기는 256 − 224 = 32이다.
이 공식을 쓰는 이유: /27의 마지막 옥텟 마스크는 이진수 11100000, 즉 224이므로 주소 범위가 32씩 나뉘기 때문이다.

계산 과정

1. /27의 서브넷 마스크는 255.255.255.224이다.
2. 마지막 옥텟은 0, 32, 64, 96, 128, 160, 192, 224를 시작점으로 32개씩 나뉜다.
3. 마지막 옥텟 253은 224~255 구간에 속하므로 네트워크 주소는 117.17.23.224이고 브로드캐스트 주소는 117.17.23.255이다.
4. 실제 호스트에 할당할 수 있는 범위는 117.17.23.225~117.17.23.254이며 총 30개이다.

NoteBase

선지 판단

① X — 네트워크 주소는 117.17.23.0이 아니라 117.17.23.224이다. 마지막 옥텟 253이 /27의 224~255 블록에 속한다. 틀린 표현 '117.17.23.0'은 '117.17.23.224'로 고쳐야 한다.

② O — /27은 마지막 옥텟의 상위 3비트까지 네트워크 비트로 사용하므로 서브넷 마스크가 255.255.255.224이다.

③ X — 클래스 기반 주소 지정에서는 첫 번째 옥텟으로 클래스를 판단한다. 117은 1~126 범위이므로 C클래스가 아니라 A클래스 주소이다. 틀린 표현 'C클래스'는 'A클래스'로 고쳐야 한다.

④ X — /27에는 전체 주소가 32개 있지만 네트워크 주소와 브로드캐스트 주소를 제외하면 사용할 수 있는 호스트 IP 주소는 30개이다. 틀린 표현 '254개'는 '30개'로 고쳐야 한다.

따라서 옳은 설명은 ②이다.

오답 포인트

①은 IP 주소의 앞 세 옥텟이 같다는 이유로 네트워크 주소를 무조건 117.17.23.0으로 보는 함정이다. /27에서는 마지막 옥텟도 네트워크 부분에 포함된다. ③은 /27이라는 현재의 접두 길이와 클래스 기반 주소 등급을 혼동하게 한다. ④의 254개는 /24에서 네트워크 주소와 브로드캐스트 주소를 제외할 때의 호스트 수이며 /27에는 적용되지 않는다.

다시 나오면 이렇게 판단

CIDR 문항은 접두 길이를 서브넷 마스크로 변환하고, 256-마지막 마스크 값으로 블록 크기를 구한 뒤, 주어진 IP가 속한 구간의 시작값을 네트워크 주소로 판단한다. 사용 가능한 호스트 수는 호스트 비트가 h일 때 $2^h - 2$로 계산한다. 클래스는 CIDR 접두 길이가 아니라 첫 번째 옥텟으로 별도 판단한다.

암기 포인트

/27은 255.255.255.224, 블록 크기 32, 전체 32개, 사용 가능한 호스트 30개이다.

NoteBase

16. <보기>에서 설명하고 있는 HTTP 프로토콜 메소드로 옳은 것은?

< 보 기 >

ㄱ. 서버로 정보를 보내는 데 사용한다.
ㄴ. 대량의 데이터를 전송할 때 사용한다.
ㄷ. 보내는 데이터가 URL을 통해 노출되지 않기 때문에 최소한의 보안성을 가진다.

① GET
② POST
③ HEAD
④ CONNECT

정답 2번

출제 포인트

서버로 데이터를 보낼 때 사용하는 HTTP 메서드의 용도와 데이터 전달 방식을 구별하는 문제이다.

핵심 개념

POST는 클라이언트가 서버에 데이터를 제출하여 처리하도록 요청하는 메서드이며, 데이터는 주로 요청 본문에 담긴다. 대량의 데이터 전송에 적합하고 데이터가 URL에 직접 표시되지 않는다는 특징이 있다.
GET은 지정한 자원을 조회하고, HEAD는 응답 본문 없이 헤더만 확인하며, CONNECT는 프록시를 거쳐 대상 서버와 터널을 설정한다.
다만 POST는 데이터가 URL에 보이지 않을 뿐 그 자체로 암호화되거나 안전해지는 것은 아니며, 전송 내용을 보호하려면 HTTPS를 사용해야 한다.

NoteBase

선지별 해설

① GET X
GET은 서버의 자원을 조회할 때 사용하는 메서드이다. 보기처럼 서버에 정보를 보내 처리하게 하고 대량의 데이터를 요청 본문에 담는 메서드라는 설명은 틀리며, 올바르게는 POST이다.

② POST O
POST는 서버에 데이터를 제출할 때 사용하며, 데이터를 URL이 아니라 주로 요청 본문에 담으므로 비교적 많은 데이터를 전송할 수 있다. 따라서 보기의 설명에 해당한다.

③ HEAD X
HEAD는 GET과 유사하지만 응답 본문을 받지 않고 응답 헤더만 확인하는 메서드이다. 보기의 데이터 제출·대량 전송 메서드라는 설명은 틀리며, 올바르게는 POST이다.

④ CONNECT X
CONNECT는 주로 프록시 서버를 통해 목적지 서버와의 터널을 설정하는 메서드이다. 보기의 일반적인 데이터 제출 메서드라는 설명은 틀리며, 올바르게는 POST이다.

오답 포인트

POST의 데이터가 URL에 드러나지 않는다는 점을 보안 보장으로 오해하기 쉽다. 이는 GET의 쿼리 문자열보다 노출이 덜할 수 있다는 의미일 뿐이며, POST 자체는 암호화 기능이 없다. 이 문항의 결정적 단서는 서버로 데이터 제출, 대량 전송, URL 비노출이다.

다시 나오면 이렇게 판단

자원 조회는 GET, 서버에 데이터 제출은 POST, 헤더만 확인하는 것은 HEAD, 프록시 터널 설정은 CONNECT로 구분한다. 데이터가 요청 본문에 담긴다는 표현이 나오면 POST를 우선 판단한다.

암기 포인트

GET은 조회, POST는 본문에 담아 제출, HEAD는 헤더만, CONNECT는 터널 설정이다.

NoteBase

17. <보기>는 대칭형 암호알고리즘이다. 이 중 국내에서 개발된 암호알고리즘을 모두 고른 것은?

< 보 기 >

ㄱ. AES	ㄴ. ARIA	ㄷ. IDEA
ㄹ. LEA	ㅁ. RC5	ㅂ. SEED

① ㄱ, ㄴ, ㄷ
② ㄱ, ㄷ, ㅁ
③ ㄴ, ㄹ, ㅂ
④ ㄹ, ㅁ, ㅂ

정답 3번

출제 포인트

대칭형 암호 알고리즘을 개발 국가에 따라 구분하여 국내에서 개발된 알고리즘을 고르는 문제이다.

핵심 개념

대칭형 암호는 암호화와 복호화에 같은 비밀키를 사용하는 방식이다.
국내 개발 알고리즘: ARIA, LEA, SEED
국외 개발 알고리즘: AES, IDEA, RC5
ARIA는 국내에서 개발된 블록 암호이고, LEA는 국내에서 개발된 경량 블록 암호이며, SEED는 한국인터넷진흥원에서 개발한 블록 암호이다. AES는 벨기에에서 개발된 Rijndael을 미국 표준으로 채택한 것이고, IDEA는 스위스에서, RC5는 미국에서 개발되었다.

선지별 해설

① X — ㄱ AES와 ㄷ IDEA는 국내 개발 알고리즘이 아니며, 국내 개발인 ㄹ LEA와 ㅂ SEED가 빠져 있다. 틀린 조합 'ㄱ, ㄴ, ㄷ'은 'ㄴ, ㄹ, ㅂ'으로 고쳐야 한다.
② X — ㄱ AES, ㄷ IDEA, ㅁ RC5는 모두 국외에서 개발된 알고리즘이다. 틀린 조합 'ㄱ, ㄷ, ㅁ'은 'ㄴ, ㄹ, ㅂ'으로 고쳐야 한다.
③ O — ㄴ ARIA, ㄹ LEA, ㅂ SEED는 모두 국내에서 개발된 대칭형 블록 암호 알고리즘이다.
④ X — ㄹ LEA와 ㅂ SEED는 국내 개발이지만, ㅁ RC5는 미국에서 개발되었고 국내 개발인 ㄴ ARIA가 빠져 있다. 틀린 조합 'ㄹ, ㅁ, ㅂ'은 'ㄴ, ㄹ, ㅂ'으로 고쳐야 한다.

NoteBase

오답 포인트

AES가 미국의 표준이라는 사실을 미국에서 개발되었다는 뜻으로 혼동하기 쉽다. AES는 벨기에에서 개발된 Rijndael을 미국 표준으로 채택한 것이므로 국내 개발 알고리즘이 아니다. 또한 RC5의 알파벳·숫자형 이름을 국내 알고리즘과 혼동하지 않아야 한다.

다시 나오면 이렇게 판단

국내 개발 대칭형 암호는 ARIA·LEA·SEED의 묶음으로 판단하고, AES·IDEA·RC5는 제외한다. 조합형 선지는 이 세 알고리즘이 모두 포함되고 다른 알고리즘은 없는지 확인한다.

암기 포인트

국산 대칭키 3종은 '아리아(ARIA)·레아(LEA)·시드(SEED)'이다.

NoteBase

18. <보기>는 서비스거부(DoS: Denial of Service) 공격 방법이다. 이 중 ICMP 프로토콜을 이용한 공격 방법으로 옳은 것의 총 개수는?

< 보 기 >

ㄱ. 랜드 공격(land attack)
ㄴ. SYN 플로딩 공격(SYN flooding attack)
ㄷ. 티어드롭 공격(teardrop attack)
ㄹ. HTTP GET 플로딩 공격(HTTP GET flooding attack)
ㅁ. 스머프 공격(smurf attack)
ㅂ. 죽음의 핑 공격(ping of death attack)

① 2개 ② 3개 ③ 4개 ④ 5개

정답 1번

출제 포인트

DoS 공격을 주로 이용하는 프로토콜에 따라 구분하고, 보기에서 ICMP를 이용하는 공격의 개수를 세는 문제이다.

핵심 개념

ICMP는 네트워크의 오류 보고와 상태 확인에 쓰이는 프로토콜이며, 에코 요청과 에코 응답은 ping에 활용된다.
ICMP 이용 공격: 스머프 공격, 죽음의 핑 공격
ICMP 이용 공격이 아닌 것: 랜드 공격과 SYN 플로딩 공격은 TCP의 SYN을 이용하고, 티어드롭 공격은 IP 단편화·재조립의 취약점을 이용하며, HTTP GET 플로딩 공격은 HTTP 요청을 대량 전송한다.
따라서 ICMP를 이용하는 공격은 ㅁ, ㅂ의 2개이다.

선지별 해설

① O — ICMP를 이용하는 것은 스머프 공격과 죽음의 핑 공격의 2개이므로 옳다.
② X — '3개'가 틀렸다. 올바르게는 ICMP를 이용하는 공격은 2개이다.
③ X — '4개'가 틀렸다. 올바르게는 ICMP를 이용하는 공격은 2개이다.
④ X — '5개'가 틀렸다. 올바르게는 ICMP를 이용하는 공격은 2개이다.

오답 포인트

티어드롭 공격을 죽음의 핑 공격과 함께 패킷 크기나 단편화 문제로 묶어 ICMP 공격으로 세기 쉽다. 그러나 티어드롭은 IP 단편의 오프셋을 비정상적으로 겹치게 하여 재조립 과정의 취약점을 노리는 공격이고, 죽음의 핑은 비정상적으로 큰 ICMP 패킷을 이용하는 공격이다.

NoteBase

다시 나오면 이렇게 판단

공격 이름마다 직접 이용하는 프로토콜을 연결한 뒤 개수를 센다. SYN은 TCP, HTTP GET은 HTTP, 단편 재조립은 IP, ping과 ICMP 에코는 ICMP로 분류한다.

암기 포인트

ICMP 공격은 스머프와 죽음의 핑, 이 문항에서는 2개이다.

NoteBase

19. 다음 글의 빈칸에 들어갈 말로 가장 적절한 것은?

Speaking two languages rather than just one has obvious practical benefits in an increasingly globalized world. However, in recent years, scientists have begun to demonstrate that the advantages of speaking more than one language are even more fundamental than the ability to converse with a wider range of people. Being _____, it turns out, makes you smarter. It can have a profound effect on your brain, improving cognitive skills that are not related to language and even shielding against dementia in old age.

① scientific ② talkative ③ bilingual ④ practical

정답 3번

출제 포인트

빈칸 앞의 '두 언어 이상을 말하는 것'과 빈칸 뒤의 뇌·인지 능력 향상이라는 내용을 연결하여 문맥에 맞는 형용사를 고르는 문제이다.

핵심 개념

bilingual은 '두 언어를 구사할 수 있는' 또는 '이중 언어를 사용하는'이라는 뜻이다. 글은 두 언어를 사용하는 사람이 더 똑똑해지고 인지 능력에서도 이점을 얻을 수 있다고 설명하므로, 'Being bilingual'이 문맥에 맞는다.

선지별 해설

① X — scientific은 '과학적인'이라는 뜻이다. 글의 scientists는 연구의 주체일 뿐, 빈칸에 들어갈 사람의 특성이 아니다. 틀린 표현 scientific은 bilingual로 고쳐야 한다.
② X — talkative는 '말이 많은, 수다스러운'이라는 뜻이다. 글은 말의 양이 아니라 두 언어를 구사하는 능력의 효과를 설명한다. 틀린 표현 talkative는 bilingual로 고쳐야 한다.
③ O — bilingual은 '두 언어를 구사하는'이라는 뜻이다. 앞에서 speaking more than one language를 반복해 받으며, '두 언어를 구사하는 것이 사람을 더 똑똑하게 만든다'라는 문장을 완성한다.
④ X — practical은 '실용적인'이라는 뜻이다. 첫 문장의 practical benefits는 이중 언어 사용의 이점을 꾸미는 표현이지, 빈칸에서 사람의 언어 사용 상태를 나타내는 말이 아니다. 틀린 표현 practical은 bilingual로 고쳐야 한다.

NoteBase

오답 포인트

①의 scientific과 ④의 practical은 각각 지문에 나온 scientists와 practical benefits 때문에 눈에 띄지만, 지문에 등장했다는 사실만으로 빈칸에 적절한 것은 아니다. Being 뒤에는 앞 문장의 '두 언어 이상을 말하는 상태'를 가리키는 bilingual이 와야 한다.

다시 나오면 이렇게 판단

빈칸이 앞 문장의 내용을 다시 받는지 먼저 확인한다. 이 문항에서는 speaking more than one language를 한 단어로 바꾸면 bilingual이 되며, 'Being bilingual makes you smarter'라는 문맥도 자연스럽다.

암기 포인트

bi-는 '둘'을 뜻하므로 bilingual은 '두 언어를 구사하는'이라는 뜻이다.

NoteBase

20. 다음 글의 빈칸 (A), (B)에 들어갈 말로 가장 적절한 것은?

The physical benefits of team sports are well-documented. Exercise is essential for maintaining a healthy weight, improving cardiovascular health, and reducing the risk of chronic diseases like heart disease, diabetes, and cancer. And what better way to exercise than by playing a sport you enjoy? When you're having fun, you're more __(A)__ to stick with it and that's when the real benefits start to show. In addition to the general health benefits of exercise, team sports also offer specific physical benefits. For example, playing soccer can help improve your balance and coordination, while basketball can help increase your vertical jump. And if you're looking to improve your __(B)__, there's no better way to do it than by playing a sport that requires you to run around for an extended period.

(A)	(B)
① liable	courage
② likely	resistance
③ liable	resistance
④ likely	endurance

정답 4번

출제 포인트

빈칸 앞뒤의 문맥을 바탕으로 형용사 likely와 명사 endurance의 의미를 구별하는 문제이다.

핵심 개념

likely: '~할 가능성이 있는'이라는 뜻으로, be likely to do는 '~할 가능성이 높다'를 나타낸다.
liable: '~할 가능성이 있는'이라는 뜻도 있지만 대체로 좋지 않은 일이 일어날 가능성이나 법적 책임을 나타낼 때 쓰인다.
endurance: 힘든 활동을 오랫동안 지속할 수 있는 '지구력'을 뜻한다.
resistance는 '저항 또는 저항력', courage는 '용기'를 뜻한다.

NoteBase

선지별 해설

① X — (A)의 liable와 (B)의 courage가 모두 문맥에 가장 적절하지 않다. 즐겁게 운동하면 계속할 가능성이 높다는 뜻에는 likely가 자연스럽고, 장시간 뛰면서 향상되는 능력은 courage가 아니라 endurance이다. 올바르게는 'likely - endurance'이다.
② X — (A)의 likely는 적절하지만 (B)의 resistance가 틀렸다. 장시간 뛰는 운동으로 기르는 것은 '저항력'이 아니라 '지구력'이므로 resistance를 endurance로 고쳐야 한다. 올바르게는 'likely - endurance'이다.
③ X — (A)의 liable와 (B)의 resistance가 모두 적절하지 않다. 이 문맥에서는 긍정적인 행동을 계속할 가능성을 나타내므로 likely가 가장 자연스럽고, 오랜 시간 뛰는 능력은 resistance가 아니라 endurance이다. 올바르게는 'likely - endurance'이다.
④ O — 'you're more likely to stick with it'은 '그 운동을 계속할 가능성이 더 높다'는 뜻이다. 또한 오랫동안 뛰어야 하는 운동은 지구력인 endurance를 향상시키므로 두 빈칸의 문맥에 모두 맞는다.

오답 포인트

liable에도 '~할 가능성이 있는'이라는 뜻이 있어 likely와 혼동하기 쉽다. 그러나 이 문장은 즐거운 운동을 계속한다는 긍정적이고 중립적인 가능성을 말하므로 likely가 더 자연스럽다. 또한 run around for an extended period는 '오랫동안 뛰다'라는 뜻이므로 resistance가 아니라 지속 능력을 뜻하는 endurance와 연결해야 한다.

다시 나오면 이렇게 판단

빈칸 하나만 따로 보지 않고 앞뒤 단서를 연결한다. 'more ___ to + 동사'가 중립적인 가능성을 나타내면 likely를 우선하고, 'for an extended period'처럼 활동의 지속 시간이 강조되면 endurance를 선택한다.

암기 포인트

be likely to는 '~할 가능성이 높다', endurance는 '오래 버티는 지구력'이다.

NoteBase