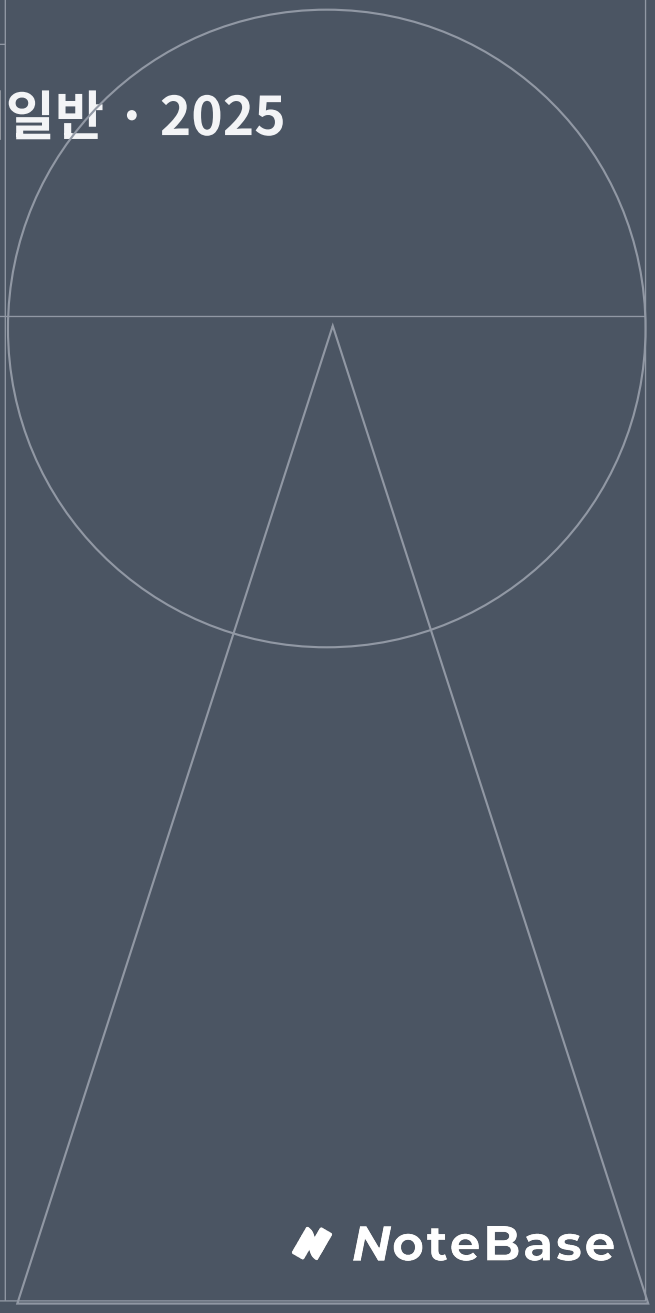
기출OX 해설집

공무원/법원직/9급 · 컴퓨터일반 · 2025



NoteBase

기출OX 해설집

공무원/법원직/9급 · 컴퓨터일반 · 2025

공무원/법원직/9급 · 컴퓨터일반 · 2025

문 1. 분산 데이터베이스 시스템의 분산 데이터 독립성을 지원하기 위해서는 분산 투명성을 보장해야 한다. 분산 투명성에 관한 다음 설명 중 가장 옳지 않은 것은?

① 위치 투명성은 데이터의 실제 저장위치를 알 필요 없이 데이터베이스의 논리적인 이름만으로 데이터에 접근할 수 있다는 것을 의미한다.

② 중복 투명성은 동일한 데이터가 여러 지역에 저장되었더라도 하나의 데이터베이스 시스템에 데이터가 저장된 것처럼 사용하는 것을 의미한다.

③ 단편화 투명성에는 수직적 단편화, 수평적 단편화, 혼합 단편화가 있다.

④ 장애 투명성은 장애가 발생하지 않도록 조치하여 데이터베이스가 운영되는 것을 의미한다.

정답 4번

출제 포인트

분산 데이터베이스의 위치·중복·단편화·장애 투명성이 각각 무엇을 사용자에게 감추는지 구별하는 문제이다.

핵심 개념

분산 투명성은 데이터가 여러 장소와 시스템에 나뉘어 있어도 사용자가 분산 구조를 의식하지 않고 하나의 데이터베이스처럼 이용하게 하는 성질이다.
위치 투명성은 저장 장소를, 중복 투명성은 복제본의 존재를, 단편화 투명성은 데이터가 조각으로 나뉜 사실을 사용자에게 감춘다. 장애 투명성은 일부 구성 요소에 장애가 발생해도 그 영향을 감추고 정상적인 데이터 접근과 처리가 이어지게 하는 성질이다.

NoteBase

선지별 해설

① O
위치 투명성이 보장되면 데이터의 물리적 저장 위치를 몰라도 논리적 이름으로 접근할 수 있다.

② O
중복 투명성이 보장되면 동일한 데이터의 복제본이 여러 지역에 있어도 사용자는 이를 하나의 논리적 데이터처럼 이용한다.

③ O
단편화 방식에는 행을 기준으로 나누는 수평적 단편화, 열을 기준으로 나누는 수직적 단편화, 두 방식을 결합한 혼합 단편화가 있다.

④ X
틀린 표현은 '장애가 발생하지 않도록 조치'한다는 부분이다. 장애 투명성은 장애 자체의 발생을 막는 개념이 아니라, 장애가 발생하더라도 그 영향을 사용자에게 감추고 시스템이 정상적으로 동작하는 것처럼 서비스를 지속하게 하는 성질이다.

오답 포인트

④는 장애에 대비한다는 일상적 의미 때문에 그럴듯하지만, 투명성은 원인을 예방하는 성질이 아니라 사용자에게 분산 구조나 장애의 영향을 드러내지 않는 성질이다.

다시 나오면 이렇게 판단

각 투명성에서 사용자에게 감추는 대상을 찾는다. 위치는 저장 장소, 중복은 복제본, 단편화는 분할 구조, 장애는 장애로 인한 영향을 감춘다.

암기 포인트

장애 투명성은 장애 예방이 아니라 장애 영향의 은폐이다.

NoteBase

문 2. 두 개의 프로세스 P1, P2가 time 0에 준비 큐(ready queue)에 P1, P2의 순서로(즉, P1이 먼저, P2가 그 다음에) 도착해 있고(즉, 두 프로세스의 도착 시간이 모두 0임), P1의 CPU 사용 시간이 80, P2의 CPU 사용 시간이 20이라 가정할 때, 시간 할당량(time quantum) 12인 RR(Round-Robin) 스케줄링 기법을 사용했을 경우 두 프로세스 모두 CPU 사용을 마칠 때까지의 평균 대기 시간(mean waiting time)은 얼마인가? (단, 문맥 교환(context switching) 시간은 0으로 하며, 두 프로세스의 입출력은 없는 것으로 한다)

① 20

② 22

③ 24

④ 26

정답 2번

출제 포인트

RR 스케줄링의 실행 순서를 시간 할당량에 따라 추적하고, 각 프로세스의 대기 시간을 구해 평균하는 문제이다.

조건 정리

도착 시각: P1과 P2 모두 0
준비 큐의 최초 순서: P1 → P2
CPU 사용 시간: P1은 80, P2는 20
시간 할당량: 12
문맥 교환 시간: 0
입출력: 없음
구해야 할 값: 두 프로세스의 평균 대기 시간

사용할 공식

입출력이 없고 도착 시각이 0이므로 각 프로세스의 대기 시간은 대기 시간 = 완료 시각 - CPU 사용 시간으로 구할 수 있다. 평균 대기 시간은 평균 대기 시간 = (P1의 대기 시간 + P2의 대기 시간)/2이다. 완료 시각은 RR 실행 순서를 시간 할당량 12씩 추적하여 구한다.

NoteBase

계산 과정

1. 0~12에는 P1이 실행되어 68이 남는다.
2. 12~24에는 P2가 실행되어 8이 남는다.
3. 24~36에는 P1이 실행되어 56이 남는다.
4. 36~44에는 P2가 남은 8을 실행하고 완료된다. 따라서 P2의 대기 시간은 $44 - 20 = 24$이다.
5. 시각 44부터는 P1만 남아 시각 100에 완료된다. 따라서 P1의 대기 시간은 $100 - 80 = 20$이다.
6. 평균 대기 시간은 $(20 + 24)/2 = 22$이다.

선지 판단

① X — 20은 P1 하나의 대기 시간일 뿐 평균 대기 시간이 아니다. ‘20’은 ‘22’로 고쳐야 한다.
② O — P1의 대기 시간 20과 P2의 대기 시간 24를 평균하면 22이다.
③ X — 24는 P2 하나의 대기 시간일 뿐 평균 대기 시간이 아니다. ‘24’는 ‘22’로 고쳐야 한다.
④ X — RR 실행 순서로 구한 두 대기 시간의 평균은 26이 아니라 22이다. ‘26’은 ‘22’로 고쳐야 한다.

오답 포인트

P1의 대기 시간 20이나 P2의 대기 시간 24를 그대로 평균 대기 시간으로 고르는 것이 함정이다. RR에서는 CPU를 실제로 사용한 시간은 대기 시간에서 제외되며, 각 프로세스의 대기 시간을 따로 구한 뒤 프로세스 수로 나눈다.

다시 나오면 이렇게 판단

먼저 시간 할당량별 실행 구간과 완료 시각을 적는다. 그다음 각 프로세스에 대해 완료 시각-도착 시각-CPU 사용 시간으로 대기 시간을 구하고, 마지막에 평균한다.

암기 포인트

대기 시간은 완료 시각-도착 시각-CPU 사용 시간이며, 평균 대기 시간은 각 프로세스의 대기 시간을 모두 더해 프로세스 수로 나눈 값이다.

NoteBase

문 3. 다음과 같은 7개의 원소를 갖는 리스트에서 이진탐색과 순차탐색을 수행하려고 한다. 다음 설명 중 가장 옳은 것은?

리스트 : {10, 11, 15, 23, 30, 58, 70}

① 평균적으로 시간복잡도는 순차탐색이 이진탐색보다 더 작다.

② 10을 검색할 때, 비교 횟수는 이진탐색은 2번, 순차탐색은 1번 비교한다.

③ 정렬이 안 된 리스트에서 순차탐색은 사용 가능하지만, 이진탐색은 사용할 수 없다.

④ 정렬이 된 리스트에서 임의의 값을 검색할 때 순차탐색의 평균 시간복잡도는 O(1)이다.

정답 3번

출제 포인트

순차탐색과 이진탐색의 전제 조건, 평균 시간복잡도, 비교 횟수의 차이를 묻는 문제이다.

핵심 개념

순차탐색은 리스트의 처음부터 원소를 하나씩 비교하는 탐색 방법으로, 정렬 여부와 관계없이 사용할 수 있고 평균 시간복잡도는 $O(n)$이다.
이진탐색은 정렬된 리스트의 중간 원소와 찾는 값을 비교하여 탐색 범위를 절반씩 줄이는 방법으로, 평균 시간복잡도는 $O(\log n)$이다.
핵심 차이는 정렬 필요 여부와 탐색 범위를 줄이는 방식이다.

NoteBase

선지별 해설

① X
평균 시간복잡도는 순차탐색이 $O(n)$, 이진탐색이 $O(\log n)$이므로 순차탐색이 더 작다는 표현이 틀렸다. 올바르게는 이진탐색의 평균 시간복잡도가 순차탐색보다 더 작다.

② X
이진탐색으로 10을 찾으면 중간값 23, 왼쪽 부분의 중간값 11, 값 10의 순서로 총 3번 비교한다. 순차탐색은 첫 원소 10과 1번 비교한다. 따라서 '이진탐색은 2번'이 틀렸으며, 올바르게는 이진탐색 3번, 순차탐색 1번이다.

③ O
순차탐색은 원소를 차례대로 확인하므로 정렬되지 않은 리스트에서도 사용할 수 있다. 반면 일반적인 이진탐색은 중간값과의 대소 관계로 탐색 범위를 절반씩 제외해야 하므로 리스트가 정렬되어 있어야 한다.

④ X
정렬된 리스트라도 순차탐색은 앞에서부터 하나씩 비교하므로 평균적으로 원소 수에 비례하는 시간이 든다. '평균 시간복잡도는 $O(1)$'이 틀렸으며, 올바르게는 평균 시간복잡도가 $O(n)$이다.

오답 포인트

②에서는 이진탐색이 항상 적은 횟수로 끝난다고 생각하기 쉽다. 이 리스트에서 10은 중간값 23과 11을 거친 뒤에야 확인되므로 비교 횟수를 실제 탐색 순서대로 세어야 한다. 또한 정렬은 이진탐색의 필수 전제이지만 순차탐색의 평균 시간복잡도를 $O(1)$로 바꾸지는 않는다.

다시 나오면 이렇게 판단

먼저 리스트의 정렬 여부를 확인한다. 정렬되어 있으면 이진탐색은 중간값을 기준으로 범위를 절반씩 줄이고, 순차탐색은 정렬 여부와 무관하게 처음부터 하나씩 확인한다. 특정 값의 비교 횟수는 각 알고리즘의 실제 비교 순서를 직접 적어 판단한다.

암기 포인트

순차탐색은 정렬 불필요·평균 $O(n)$, 이진탐색은 정렬 필요·평균 $O(\log n)$이다.

NoteBase

문 4. 다음 중 TCP/IP 프로토콜의 인터넷 계층에서 수행하는 프로토콜에 해당하지 않는 것은?

① ICMP

② ARP

③ FDDI

④ IP

정답 3번

출제 포인트

TCP/IP의 인터넷 계층에서 동작하는 프로토콜과 네트워크 액세스 계층의 표준을 구분하는 문제이다.

핵심 개념

인터넷 계층은 IP 주소를 바탕으로 패킷을 전달하고 경로를 선택하는 계층이다. IP가 핵심 프로토콜이며, ICMP는 오류·상태 정보를 전달하고 ARP는 IPv4 주소에 대응하는 MAC 주소를 알아낸다. FDDI는 광섬유 기반의 LAN 전송 표준으로서 인터넷 계층 프로토콜이 아니라 네트워크 액세스 계층에 해당한다.

선지별 해설

① O — ICMP는 IP 패킷 전달 과정에서 발생한 오류와 네트워크 상태를 알리는 인터넷 계층의 제어 메시지 프로토콜이다.
② O — ARP는 같은 네트워크에서 목적지의 IPv4 주소에 대응하는 MAC 주소를 알아내는 프로토콜이며, TCP/IP 계층 분류에서는 인터넷 계층 관련 프로토콜로 다룬다.
③ X — FDDI를 인터넷 계층 프로토콜로 본 부분이 틀렸다. 올바르게는 FDDI는 광섬유를 전송 매체로 사용하는 LAN 표준이며 네트워크 액세스 계층에 해당한다.
④ O — IP는 논리 주소 지정과 패킷 전달 및 라우팅을 담당하는 인터넷 계층의 핵심 프로토콜이다.

오답 포인트

ARP가 MAC 주소를 다룬다는 이유만으로 FDDI와 같은 계층이라고 판단하기 쉽다. 이 문항의 TCP/IP 계층 분류에서는 ARP를 인터넷 계층 관련 프로토콜로 보고, 실제 LAN 전송 방식인 FDDI를 네트워크 액세스 계층으로 구분한다.

다시 나오면 이렇게 판단

IP 주소 기반 전달·제어·주소 해석에 관여하는 IP, ICMP, ARP와 실제 네트워크 매체 및 LAN 접속 방식을 정하는 FDDI를 구분한다.

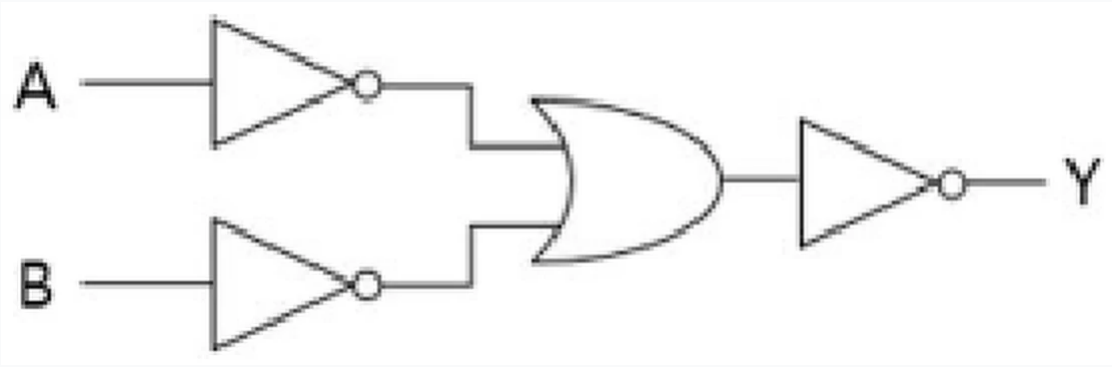
NoteBase

암기 포인트

인터넷 계층은 IP·ICMP·ARP, 네트워크 액세스 계층은 FDDI이다.

NoteBase

문 5. 다음 논리회로의 논리식 Y는?



① $Y = A + B$

② $Y = A \cdot B$

③ $Y = A' + B'$

④ $Y = A' \cdot B'$

정답 2번

출제 포인트

입력 A와 B는 각각 NOT 게이트를 거쳐 A'와 B'가 되고, 두 값이 OR 게이트를 지난 뒤 다시 NOT 게이트를 거친다. 따라서 출력은 드모르간 법칙에 따라 $Y = (A' + B')' = A \cdot B$가 된다.

NoteBase

선지별 해설

① X

$Y = A + B$는 A와 B를 그대로 OR 연산한 식이다. 이 회로에는 입력의 두 NOT 게이트와 출력의 NOT 게이트가 있으므로 해당 식이 아니다. 올바르게는 $Y = (A' + B')' = A \cdot B$이다.

② O

각 입력을 반전한 뒤 OR 연산하고 그 결과를 다시 반전하므로 $Y = (A' + B')'$이다. 드모르간 법칙을 적용하면 $Y = A \cdot B$가 된다.

③ X

$Y = A' + B'$는 가운데 OR 게이트의 출력까지만 나타낸 식으로, 마지막 NOT 게이트를 반영하지 않았다. 올바르게는 $Y = (A' + B')' = A \cdot B$이다.

④ X

$Y = A' \cdot B'$는 드모르간 법칙에 따라 $(A + B)'$와 같은 식이며 이 회로의 연결과 다르다. 올바르게는 $Y = (A' + B')' = A \cdot B$이다.

오답 포인트

③의 $A' + B'$를 최종 출력으로 착각하기 쉽다. 이는 가운데 OR 게이트 직후의 값일 뿐이며, 회로 오른쪽의 마지막 NOT 게이트까지 적용해야 전체 출력 Y가 된다.

다시 나오면 이렇게 판단

입력에서 출력 방향으로 각 게이트의 중간 식을 차례로 적는다. 이 회로에서는 $A', B' \rightarrow A' + B' \rightarrow (A' + B')'$ 순서로 정리한 뒤 드모르간 법칙을 적용한다.

암기 포인트

반전된 두 입력을 OR한 뒤 다시 반전하면 AND가 된다: $(A' + B')' = A \cdot B$.

NoteBase

문 6. 다음 C 프로그램의 결과로 옳은 것은?

```
#include <stdio.h>
void sub(int x, int *y);
int main(void)
{
        int a = 1, b = 5;
        sub(a, &b);
        printf("%d, %d\n", a, b);
}
void sub(int x, int *y)
{
        x = x + 2;
        *y = *y + x;
}
```

① 1, 5

② 1, 8

③ 3, 5

④ 3, 8

정답 2번

출제 포인트

이 문제는 C 함수 호출에서 일반 정수 매개변수의 값 전달과 포인터 매개변수를 통한 원본 값 변경의 차이를 묻는다.

핵심 개념

값 전달: sub(a, &b)에서 a의 값 1이 매개변수 x에 복사되므로 x를 3으로 바꾸어도 main의 a는 1로 유지된다.
포인터 전달 효과: &b는 b의 주소이고 y가 그 주소를 받으므로 *y를 바꾸면 main의 b가 직접 바뀐다.
실행 결과: x = x + 2로 x는 3이 되고, *y = *y + x로 b는 5 + 3 = 8이 된다. 따라서 printf가 출력하는 a, b는 1, 8이다.

NoteBase

선지별 해설

① X — ‘1, 5’에서 b가 5로 유지된다는 부분이 틀리다. y가 b의 주소를 가리키므로 *y = *y + x 실행 후 b는 8이 되며, 올바른 출력은 1, 8이다.
② O — a는 값으로 전달되어 1을 유지하고, b는 포인터 y를 통해 8로 변경되므로 1, 8이 출력된다.
③ X — ‘3, 5’는 a와 b를 모두 잘못 판단한 결과이다. x가 3이 되어도 a는 1이고, *y의 변경으로 b는 8이므로 올바른 출력은 1, 8이다.
④ X — ‘3, 8’에서 a가 3이 된다는 부분이 틀리다. 3이 되는 것은 지역 매개변수 x이며 main의 a는 1을 유지하므로 올바른 출력은 1, 8이다.

오답 포인트

x = x + 2를 보고 main의 a도 3이 된다고 착각하기 쉽다. x는 a의 복사본이지만, *y는 b의 주소가 가리키는 원본이므로 두 대입문의 영향 범위가 다르다.

다시 나오면 이렇게 판단

함수 호출 인수마다 값 자체가 복사되는지 주소가 전달되는지를 먼저 구분한다. 일반 매개변수의 변경은 호출한 함수의 변수에 영향을 주지 않지만, 전달받은 주소를 역참조한 변경은 해당 원본 변수에 반영된다.

암기 포인트

x는 a의 복사본, *y는 b의 원본이다.

NoteBase

문 7. 어느 시스템에 하나의 자원형(resource type)만이 존재하고 이에 5개의 단위자원(resource units)이 있으며, 이 시스템에 p개의 프로세스가 존재한다고 가정할 때, 각 프로세스들이 어느 순간에도 최대 2개까지의 단위 자원만을 필요로 한다면 이 시스템에서 교착상태의 발생을 근본적으로 없애기 위한 필요충분조건은?

① $p \leq 2$

② $p \leq 3$

③ $p \leq 4$

④ $p \leq 5$

정답 3번

출제 포인트

동일한 자원형의 단위자원이 여러 개 있을 때, 모든 프로세스가 최대 필요량까지 자원을 확보할 수 있도록 하여 교착상태를 원천적으로 배제하는 조건을 구하는 문제이다.

조건 정리

단위자원의 총수는 5개이고, 프로세스 수는 p개이며, 각 프로세스의 최대 필요량은 2개이다. 교착상태가 발생하지 않으려면 각 프로세스가 최대 필요량보다 1개 적은 자원을 보유한 최악의 상황에서도 적어도 1개의 여유 자원이 남아야 한다.

사용할 공식

동일한 자원형에서 단위자원 수를 R, 프로세스 수를 p, 각 프로세스의 최대 필요량을 k라고 하면 교착상태를 원천적으로 배제하는 조건은 다음과 같다.

$R \geq p(k-1)+1$

각 프로세스가 $k-1$개씩 보유해도 1개가 남으면, 그 자원을 한 프로세스에 주어 작업을 끝내게 할 수 있고 반환된 자원으로 나머지 프로세스도 차례로 종료할 수 있기 때문이다.

계산 과정

주어진 $R=5, k=2$를 공식에 대입한다.

$5 \geq p(2-1)+1 = p+1$

따라서 $p \leq 4$이다. 반대로 $p=5$이면 5개 프로세스가 자원을 1개씩 보유한 채 각각 두 번째 자원을 기다릴 수 있으므로 교착상태가 발생할 수 있다.

NoteBase

선지 판단

① X — $p \le 2$이면 교착상태는 방지되지만, $p = 3$과 $p = 4$도 안전하므로 필요조건이 되지 못한다. 올바르게는 $p \le 4$이다.
② X — $p \le 3$은 충분히 안전하지만 $p = 4$까지도 교착상태를 원천적으로 배제할 수 있으므로 필요충분조건이 아니다. 올바르게는 $p \le 4$이다.
③ O — 계산 결과와 같은 $p \le 4$이며, 안전한 프로세스 수의 최대 범위를 정확히 나타낸다.
④ X — $p \le 5$에는 위험한 경우인 $p = 5$가 포함된다. 프로세스 5개가 자원을 1개씩 보유하고 추가 자원 1개씩을 기다리면 남은 자원이 없어 교착상태가 가능하다. 올바르게는 $p \le 4$이다.

오답 포인트

①과 ②는 교착상태를 막는 충분조건이지만 범위를 지나치게 좁혀 필요조건이 되지 못한다. 문제는 단순히 안전한 조건이 아니라 필요충분조건을 묻고 있으므로, 안전성이 보장되는 최대 프로세스 수까지 포함해야 한다.

다시 나오면 이렇게 판단

각 프로세스에 최대 필요량보다 1개 적게 배분한 뒤에도 자원 1개가 남는지를 확인한다. 즉 $R \ge p(k-1)+1$을 세우고, 경계값을 하나 초과한 경우 실제로 교착상태 배치가 가능한지도 검산한다.

암기 포인트

교착상태 원천 배제 조건은 '모두가 최대치보다 1개 적게 가진 뒤에도 1개가 남는다', 즉 $R \ge p(k-1)+1$이다.

NoteBase

문 8. 현재 A, B, C 프로그램을 실행 중이며, 이 세 개의 프로그램으로 주메모리가 꽉 차 있는 상태이다. 이 상태에서 새로운 프로그램 D를 실행하려고 시도할 때, 다음 설명 중 가장 옳은 것은?

① 메모리 부족으로 인해 프로그램 D는 실행이 거부된다.

② 프로그램 D는 주메모리가 확보될 때까지 대기 상태로 유지된다.

③ 운영체제는 가상메모리를 사용하여 부족한 주메모리를 보완하고 D를 실행한다.

④ 운영체제는 실행 중인 프로그램 중 가장 오래 전에 사용된 하나를 강제로 종료시킨 후 D를 실행한다.

정답 3번

출제 포인트

주메모리가 가득 찼을 때 운영체제가 가상메모리와 페이지 교체를 이용해 새 프로그램을 실행할 수 있는지를 묻는 문제이다.

핵심 개념

가상메모리는 보조기억장치의 일부를 주메모리처럼 활용하여 물리적 주메모리보다 큰 주소 공간을 제공하는 메모리 관리 기법이다. 주메모리에 빈 공간이 없으면 운영체제는 당장 필요하지 않은 페이지를 보조기억장치로 내보내고 필요한 페이지를 적재하는 페이지 교체를 수행할 수 있다. 프로그램 자체를 종료하는 것이 아니라 프로그램을 구성하는 페이지의 적재 상태를 조정한다.

선지별 해설

① X — 주메모리가 가득 찼다는 이유만으로 D의 실행이 곧바로 거부되는 것은 아니다. 틀린 표현은 '실행이 거부된다'이며, 올바르게는 운영체제가 가상메모리와 페이지 교체를 이용해 D 실행에 필요한 공간을 마련할 수 있다.

② X — D가 주메모리 전체의 빈 공간이 생길 때까지 반드시 대기하는 것은 아니다. 틀린 표현은 '주메모리가 확보될 때까지 대기 상태로 유지된다'이며, 올바르게는 필요한 페이지만 주메모리에 적재하고 기존 페이지 일부를 교체하여 D를 실행할 수 있다.

③ O — 운영체제는 가상메모리를 이용하여 사용 빈도가 낮거나 당장 필요하지 않은 페이지를 보조기억장치로 옮기고, 확보된 주메모리 공간에 D의 필요한 페이지를 적재하여 실행할 수 있다.

④ X — 페이지 교체와 프로그램 강제 종료는 다르다. 틀린 표현은 '가장 오래 전에 사용된 하나를 강제로 종료'이며, 올바르게는 교체 정책에 따라 페이지를 주메모리에서 내보낼 수 있을 뿐 해당 프로그램 전체를 강제로 종료하지 않는다.

NoteBase

오답 포인트

④의 '가장 오래 전에 사용된'이라는 표현은 LRU 페이지 교체와 혼동하게 만든다. LRU가 교체하는 대상은 프로그램 전체가 아니라 가장 오랫동안 사용되지 않은 페이지이다. 또한 주메모리가 가득 찬 상태를 곧바로 실행 거부나 무조건적인 대기로 연결하면 ①이나 ②를 고르게 된다.

다시 나오면 이렇게 판단

주메모리 부족 상황에서는 먼저 가상메모리의 페이지 교체 가능성을 확인한다. 운영체제는 프로세스 전체를 종료하기보다 불필요한 페이지를 보조기억장치로 내보내고 새로 필요한 페이지를 적재한다.

암기 포인트

메모리가 가득 차면 프로그램 종료가 아니라 페이지 교체가 우선이다.

NoteBase

문 9. HTTPS가 HTTP보다 안전한 이유에 대한 설명으로 가장 옳은 것은?

① 포트 번호가 다르기 때문이다.

② 텍스트 파일로 작동하기 때문이다.

③ 서버의 응답 속도가 더 빠르기 때문이다.

④ 인증서를 사용하여 암호화된 통신을 제공하기 때문이다.

정답 4번

출제 포인트

HTTPS가 HTTP보다 안전한 핵심 이유가 단순한 포트 차이가 아니라 TLS를 통한 인증·암호화·무결성 보호에 있음을 묻는 문제이다.

핵심 개념

HTTP는 웹 클라이언트와 서버가 요청과 응답을 주고받는 통신 규약이다. HTTPS는 HTTP 통신에 TLS 보안을 적용하여 인증서로 서버의 신원을 확인하고, 전송 내용을 암호화하며, 통신 중 데이터가 변조되었는지도 확인할 수 있게 한다.

선지별 해설

① X — 포트 번호가 다르다는 사실 자체는 통신 내용을 보호하지 않는다. 일반적으로 HTTP와 HTTPS가 서로 다른 기본 포트를 사용하지만, 올바르게는 TLS가 제공하는 인증·암호화·무결성 보호 때문에 HTTPS가 더 안전하다.
② X — '텍스트 파일로 작동하기 때문'이라는 표현은 HTTPS의 안전성과 관련이 없다. 올바르게는 HTTP 메시지를 TLS로 보호하여 네트워크에서 내용이 그대로 노출되지 않게 한다.
③ X — 서버의 응답 속도는 보안성의 근거가 아니다. 올바르게는 인증서와 TLS 암호화로 상대방을 확인하고 전송 데이터를 보호하기 때문에 더 안전하다.
④ O — HTTPS는 인증서를 활용한 TLS 통신으로 서버 인증과 암호화된 데이터 전송을 제공하므로 가장 옳은 설명이다.

오답 포인트

①의 '포트 번호가 다르기 때문'은 HTTP와 HTTPS의 겉으로 드러나는 차이를 보안의 원인으로 바꿔 놓은 함정이다. ③의 속도 역시 성능의 문제일 뿐 보안의 판단 기준이 아니다.

다시 나오면 이렇게 판단

HTTPS의 안전성을 묻는 문항에서는 포트나 속도가 아니라 인증서, TLS, 암호화, 무결성 보호가 언급된 선지를 찾는다.

NoteBase

암기 포인트

HTTPS는 HTTP에 TLS를 더한 것으로, 핵심은 인증·암호화·무결성이다.

NoteBase

문 10. 다음 중 클래스 설계 원칙에 해당하지 않는 것은?

① 의존성 역전의 원칙

② 인터페이스 통합의 원칙

③ 개방 폐쇄의 원칙

④ 리스코프 교체의 원칙

정답 2번

출제 포인트

객체지향 클래스 설계의 SOLID 원칙에 포함되는 명칭과 포함되지 않는 명칭을 구별하는 문제이다.

핵심 개념

SOLID는 단일 책임 원칙, 개방·폐쇄 원칙, 리스코프 교체 원칙, 인터페이스 분리 원칙, 의존성 역전 원칙의 머리글자를 묶은 클래스 설계 원칙이다. 핵심은 '인터페이스 통합'이 아니라 클라이언트가 사용하지 않는 기능에 의존하지 않도록 인터페이스를 작게 나누는 '인터페이스 분리'이다.

선지별 해설

① O — 의존성 역전의 원칙은 상위 모듈과 하위 모듈이 구체 구현이 아니라 추상화에 의존해야 한다는 SOLID 원칙이다.
② X — 틀린 표현은 '인터페이스 통합의 원칙'이다. SOLID에 해당하는 올바른 명칭은 '인터페이스 분리의 원칙'이며, 클라이언트가 사용하지 않는 메서드에 의존하지 않도록 인터페이스를 분리한다.
③ O — 개방·폐쇄의 원칙은 소프트웨어 요소가 확장에는 열려 있고 기존 코드의 변경에는 닫혀 있어야 한다는 SOLID 원칙이다.
④ O — 리스코프 교체의 원칙은 하위 타입의 객체가 상위 타입의 객체를 대신해도 프로그램의 올바른 동작이 유지되어야 한다는 SOLID 원칙이다.

오답 포인트

②의 '통합'은 그럴듯하지만 SOLID의 I는 Interface Segregation, 즉 인터페이스 '분리'를 뜻한다. 통합과 분리를 반대로 바꾼 명칭이 함정이다.

다시 나오면 이렇게 판단

SOLID의 다섯 머리글자를 단일 책임(S), 개방·폐쇄(O), 리스코프 교체(L), 인터페이스 분리(I), 의존성 역전(D)에 대응시켜 선지의 명칭을 확인한다.

NoteBase

암기 포인트

SOLID의 I는 통합이 아니라 분리이다.

NoteBase

문 11. 인터럽트(interrupt)에 관한 다음 설명 중 가장 옳지 않은 것은?

① 인터럽트가 발생함으로 인하여 문맥 교환(context switching)이 이루어질 수 있다.

② 인터럽트가 발생하더라도 CPU에서 실행 중이던 기계어 명령은 끝까지 실행된다.

③ 인터럽트 핸들러(interrupt handler)의 실행은 커널 모드(kernel mode)에서 진행된다.

④ 인터럽트 처리가 종료되면 항상 인터럽트 발생 당시 실행 중이던 프로세스가 다시 실행된다.

정답 4번

출제 포인트

인터럽트가 발생한 뒤 CPU가 현재 명령을 마치고 제어권을 커널의 인터럽트 핸들러로 넘기는 과정과, 처리 후 실행할 프로세스가 스케줄링에 따라 달라질 수 있음을 묻는 문제이다.

핵심 개념

인터럽트는 하드웨어나 소프트웨어의 사건을 CPU에 알려 현재 실행 흐름을 잠시 중단시키고 정해진 처리 루틴을 수행하게 하는 신호이다. 일반적인 인터럽트 처리에서는 현재 명령의 실행을 완료한 뒤 CPU 상태를 저장하고 커널 모드의 인터럽트 핸들러를 실행한다. 처리가 끝난 뒤에는 스케줄러의 판단에 따라 원래 프로세스가 계속 실행될 수도 있고 다른 프로세스가 실행될 수도 있다.

선지별 해설

① O — 인터럽트 처리로 커널에 제어권이 넘어간 뒤 스케줄러가 다른 프로세스를 선택하면, 실행 중인 프로세스의 상태를 저장하고 새 프로세스의 상태를 복원하는 문맥 교환이 이루어질 수 있다.
② O — 일반적인 인터럽트는 CPU가 현재 실행 중인 기계어 명령을 완료한 명령어 경계에서 인식·처리되므로, 실행 중이던 명령은 끝까지 수행된다.
③ O — 인터럽트 핸들러는 장치 제어와 시스템 상태 처리 등 특권이 필요한 커널 코드이므로 커널 모드에서 실행된다.
④ X — 틀린 표현은 '항상 인터럽트 발생 당시 실행 중이던 프로세스가 다시 실행된다'이다. 올바르게는 인터럽트 처리 후 스케줄링 결과에 따라 원래 프로세스가 재개될 수도 있고, 우선순위가 더 높은 프로세스 등 다른 프로세스가 실행될 수도 있다.

오답 포인트

인터럽트 처리가 끝나면 중단 지점으로 복귀할 수 있다는 사실을 원래 프로세스가 반드시 즉시 재개된다는 뜻으로 오해하기 쉽다. ④의 '항상'이 함정이며, 복귀할 실행 대상은 스케줄링 결과에 따라 달라진다.

NoteBase

다시 나오면 이렇게 판단

인터럽트 문항에서는 '현재 명령 완료 → 상태 저장 → 커널 모드에서 핸들러 실행 → 복귀 또는 스케줄링'의 흐름으로 판단한다. '항상 원래 프로세스가 실행된다'처럼 스케줄러의 선택 가능성을 배제하는 단정은 틀린 표현이다.

암기 포인트

인터럽트는 명령 완료 뒤 처리하고, 핸들러는 커널 모드에서 실행하며, 처리 후 실행 대상은 스케줄러가 정한다.

NoteBase

문 12. 인터넷의 주소 체계에 관한 다음 설명 중 가장 옳지 않은 것은?

① DNS(Domain Name System)는 계층적 이름 체계를 기반으로 IP 주소를 관리한다.

② DNS(Domain Name System)는 IP 주소를 문자로 된 도메인 네임으로 변환하는 역할을 한다.

③ IPv4 주소는 32비트 주소 체계로, 네트워크 부분과 호스트 부분의 길이에 따라 A~E 클래스 구조를 갖는다.

④ IPv6 주소는 IPv4의 주소 부족 문제를 해결하기 위해 개발되었으며, 콜론(:)으로 구분된 128비트 주소 체계를 사용한다.

정답 2번

출제 포인트

DNS의 기본 변환 방향과 IPv4·IPv6 주소 체계의 특징을 구별하는 문제이다.

핵심 개념

DNS: 계층적인 도메인 이름 공간과 분산 데이터베이스를 이용해 도메인 네임을 IP 주소 등의 자원 정보에 대응시키는 시스템이다. 기본적인 정방향 조회는 도메인 네임 → IP 주소이다.
IPv4: 32비트 주소 체계이며, 전통적인 클래스 기반 방식에서는 A~E 클래스로 구분한다.
IPv6: IPv4의 주소 부족을 보완하기 위한 128비트 주소 체계이며, 16진수 그룹을 콜론(:)으로 구분한다.

선지별 해설

① O — DNS는 최상위 도메인, 하위 도메인 등으로 이어지는 계층적 이름 체계를 사용하며, 주소 레코드를 통해 도메인 네임과 IP 주소의 대응 관계를 관리한다.
② X — 틀린 부분은 'IP 주소를 문자로 된 도메인 네임으로 변환'이다. DNS의 대표적인 역할은 문자로 된 도메인 네임을 IP 주소로 변환하는 정방향 조회이다. IP 주소에서 도메인 네임을 찾는 것은 PTR 레코드를 이용하는 역방향 조회에 해당한다.
③ O — IPv4 주소의 길이는 32비트이다. 전통적인 클래스 기반 주소 방식은 선두 비트에 따라 A~E 클래스로 구분하며, A~C 클래스는 네트워크 부분과 호스트 부분의 경계가 서로 다르다. D 클래스는 멀티캐스트용, E 클래스는 예약·실험용이다.
④ O — IPv6는 IPv4의 주소 공간 부족을 해결하기 위해 128비트 주소를 사용하며, 16진수로 표시한 주소 그룹을 콜론(:)으로 구분한다.

오답 포인트

②는 DNS의 변환 방향을 반대로 제시한 선지이다. IP 주소로 도메인 네임을 찾는 역방향 조회도 존재하지만, DNS의 기본 역할을 묻는 문맥에서는 도메인 네임을 IP 주소로 변환하는 정방향 조회를 기준으로 판단한다.

NoteBase

다시 나오면 이렇게 판단

DNS 선지는 변환 전후의 대상을 화살표로 바꾸어 확인한다. 정방향 조회는 도메인 네임 → IP 주소이고, 역방향 조회는 IP 주소 → 도메인 네임이다. IP 주소 체계는 IPv4 32비트, IPv6 128비트와 콜론 구분을 기준으로 판별한다.

암기 포인트

DNS 정방향은 이름 → IP, IPv4는 32비트, IPv6는 128비트·콜론이다.

NoteBase

문 13. 다음 프로세스 스케줄링 기법 중 순환 순서 기법(Round Robin)의 설명에 해당하는 것은?

① 가장 먼저 들어온 프로세스를 가장 먼저 처리한다.

② 자원을 가장 적게 사용하는 프로세스를 가장 먼저 처리한다.

③ CPU 실행시간이 가장 짧은 프로세스를 가장 먼저 실행하면서 순서대로 처리한다.

④ 준비상태에 있는 모든 프로세스에게 CPU 실행시간을 골고루 돌아가면서 나누어 주며 실행한다.

정답 4번

출제 포인트

순환 순서 기법(Round Robin)이 프로세스마다 일정한 CPU 시간 할당량을 차례로 제공하는 선점형 스케줄링이라는 점을 묻는 문제이다.

핵심 개념

Round Robin: 준비 큐의 각 프로세스에 일정한 시간 할당량을 차례로 부여하고, 그 시간 안에 끝나지 않은 프로세스는 중단하여 준비 큐의 뒤로 보내는 선점형 스케줄링이다.
FCFS는 도착 순서, SJF는 예상 CPU 실행시간이 짧은 순서, Round Robin은 일정한 시간 할당량을 순환하며 배정하는 방식이다.

선지별 해설

① X — '가장 먼저 들어온 프로세스를 가장 먼저 처리한다'는 FCFS의 설명이다. Round Robin은 준비 큐의 각 프로세스에 일정한 CPU 시간 할당량을 차례로 배정한다.
② X — '자원을 가장 적게 사용하는 프로세스'를 우선한다는 기준은 Round Robin의 기준이 아니다. Round Robin은 자원 사용량과 관계없이 준비 큐의 프로세스들을 순환하면서 일정한 CPU 시간 할당량을 준다.
③ X — 'CPU 실행시간이 가장 짧은 프로세스를 가장 먼저 실행한다'는 SJF의 설명이다. Round Robin은 실행시간의 장단으로 우선순위를 정하지 않고 일정한 시간 할당량을 차례로 제공한다.
④ O — 준비상태의 모든 프로세스에 CPU 시간을 일정한 시간 할당량으로 나누어 돌아가며 실행한다는 설명은 Round Robin의 핵심 특징과 일치한다.

오답 포인트

①의 '먼저 들어온'은 FCFS, ③의 '실행시간이 가장 짧은'은 SJF를 가리킨다. Round Robin을 단순히 순서대로 처리하는 방식으로만 보면 ①과 혼동하기 쉽지만, 핵심은 프로세스마다 일정한 시간 할당량을 반복하여 제공한다는 점이다.

NoteBase

다시 나오면 이렇게 판단

선지에서 ‘일정한 시간 할당량’, ‘차례로 순환’, ‘시간이 끝나면 준비 큐 뒤로 이동’이라는 단서를 확인하면 Round Robin으로 판단한다.

암기 포인트

FCFS는 도착 순서, SJF는 짧은 작업 우선, Round Robin은 시간 할당량을 돌아가며 배정한다.

NoteBase

문 14. 분산 데이터베이스 시스템에서 교착상태 예방(deadlock prevention)을 위하여 wound-wait 기법을 사용한다고 가정할 때, 이 시스템에 트랜잭션 T1과 T2가 존재하고, T1의 우선순위(priority)가 T2보다 높은 경우 T2가 할당받은 자원을 T1이 요청한다면 어떤 결과가 발생하는가?

① T1은 T2가 해당 자원을 반납할 때까지 대기 상태에서 기다린다.

② T2가 할당받고 있는 자원을 선점(preemption)하여 T1에게 할당한다.

③ T1의 실행을 취소(abort)하고 이후 재실행(restart)하도록 한다.

④ T1으로 하여금 busy-waiting 하면서 해당 자원의 점유를 시도하게 한다.

정답 2번

출제 포인트

wound-wait 기법에서 우선순위가 높은 트랜잭션이 낮은 트랜잭션이 보유한 자원을 요청할 때의 처리 방식을 묻는 문제이다.

핵심 개념

wound-wait는 교착상태를 예방하기 위해 트랜잭션의 우선순위를 이용하는 방식이다. 우선순위가 높은 트랜잭션이 낮은 트랜잭션의 자원을 요청하면 낮은 트랜잭션을 중단·롤백시켜 자원을 회수하고, 반대로 우선순위가 낮은 트랜잭션이 높은 트랜잭션의 자원을 요청하면 낮은 트랜잭션이 기다린다.

선지별 해설

① X — T1이 기다린다는 부분이 틀리다. T1의 우선순위가 더 높으므로 T1이 대기하는 것이 아니라 T2를 중단·롤백시켜 자원을 회수한다.
② O — 우선순위가 높은 T1이 낮은 T2를 wound하여 T2가 보유한 자원을 회수하고 T1에 할당한다. 이때 T2는 롤백된 뒤 나중에 재시작한다.
③ X — T1의 실행을 취소한다는 부분이 틀리다. 취소·롤백되는 것은 우선순위가 낮고 자원을 보유한 T2이며, T1은 회수된 자원을 할당받는다.
④ X — busy-waiting으로 반복 점유를 시도한다는 부분이 틀리다. wound-wait는 우선순위에 따라 상대 트랜잭션을 롤백시키거나 요청 트랜잭션을 대기시키는 방식이다.

오답 포인트

①의 ‘T1이 대기한다’와 ③의 ‘T1을 취소한다’처럼 요청자의 우선순위를 거꾸로 적용하기 쉽다. 이 상황에서는 높은 우선순위의 T1이 낮은 우선순위의 T2를 중단시키는 것이 핵심이다.

NoteBase

다시 나오면 이렇게 판단

요청 트랜잭션과 자원 보유 트랜잭션의 우선순위를 먼저 비교한다. 요청자가 더 높으면 보유자를 롤백시키고, 요청자가 더 낮으면 요청자가 대기한다.

암기 포인트

wound-wait는 '높은 우선순위가 낮은 우선순위를 wound하여 롤백, 낮은 우선순위는 높은 우선순위를 wait'로 기억한다.

NoteBase

문 15. 페이징(paging) 기법을 사용하는 가상 메모리(virtual memory) 시스템에 관한 다음 설명 중 가장 옳지 않은 것은?

① 페이지 프레임(page frame) 크기는 페이지(page) 크기와 같다.

② 가상 주소(virtual address)를 실 주소(real address)로 변환하는 과정의 성능 개선을 위해 TLB(Translation Lookaside Buffer)를 사용할 수 있다.

③ 페이지 폴트(page fault)는 문맥 교환(context switching)을 유발시킨다.

④ 역전 페이지 테이블(inverted page table)을 사용할 경우 주소 변환 시간 측면에서 성능을 개선시킬 수 있다.

정답 4번

출제 포인트

페이징에서 페이지와 페이지 프레임의 관계, TLB의 역할, 페이지 폴트의 처리, 역전 페이지 테이블의 장단점을 구별하는 문제이다.

핵심 개념

페이징은 가상 주소 공간을 고정 크기의 페이지로, 물리 메모리를 같은 크기의 페이지 프레임으로 나누어 대응시키는 기법이다.
TLB는 최근 사용한 페이지 테이블 항목을 보관하여 주소 변환을 빠르게 하는 고속 캐시이다.
역전 페이지 테이블은 물리 페이지 프레임마다 하나의 항목을 두어 페이지 테이블의 메모리 사용량을 줄이는 구조이다. 다만 가상 페이지에 대응하는 항목을 찾는 과정이 필요하므로 주소 변환시간 자체를 단축하는 구조는 아니다.

선지별 해설

① O — 페이지는 가상 주소 공간의 블록이고 페이지 프레임은 물리 메모리의 블록이며, 서로 적재될 수 있도록 두 크기는 같다.
② O — TLB에 필요한 페이지 테이블 항목이 있으면 주기억장치의 페이지 테이블을 다시 조회하지 않고 주소를 변환할 수 있어 성능이 향상된다.
③ O — 페이지 폴트가 발생하면 운영체제가 디스크에서 해당 페이지를 적재하는 동안 현재 프로세스가 대기 상태가 되고 다른 프로세스가 실행될 수 있으므로 문맥 교환이 유발된다.
④ X — 틀린 표현은 '주소 변환시간 측면에서 성능을 개선시킬 수 있다'이다. 역전 페이지 테이블의 주된 장점은 페이지 테이블의 메모리 공간 절약이며, 주소 변환에는 항목 탐색이 필요해 시간이 더 걸릴 수 있다. 올바르게는 '역전 페이지 테이블을 사용하면 페이지 테이블이 차지하는 메모리 공간을 줄일 수 있다'이다.

NoteBase

오답 포인트

④에서 ‘성능 개선’을 모든 자원에 대한 장점으로 넓혀 해석하기 쉽다. 역전 페이지 테이블은 공간 효율을 높이지만 주소 변환 속도까지 자동으로 높이지는 않는다.

다시 나오면 이렇게 판단

TLB는 주소 변환 속도, 역전 페이지 테이블은 페이지 테이블의 공간 절약이라는 핵심 목적을 먼저 구분한다. 페이지 폴트는 보조기억장치 접근과 프로세스 대기를 동반하므로 문맥 교환 가능성과 연결한다.

암기 포인트

TLB는 시간 절약, 역전 페이지 테이블은 공간 절약이다.

NoteBase

문 16. 다음 설명 중 가장 옳지 않은 것은?

① 커널의 역할 중 프로세스 관리는 프로세스에 CPU를 배분하고, 프로세스가 잘 수행하도록 작업공간을 배치하며, 실제 메모리보다 큰 가상공간을 제공한다.

② 커널은 단일형 구조 커널, 계층형 구조 커널, 마이크로 구조 커널로 구분할 수 있다.

③ 운영체제는 커널과 인터페이스로 구성되어 있는데, 인터페이스는 사용자와 응용 프로그램에 인접하여 실행 결과를 사용자와 응용 프로그램에 돌려준다.

④ 단일형 구조 커널은 초창기 운영체제의 구조로 모듈이 거의 분리되지 않았기 때문에 모듈간의 통신비용이 줄어 효율적인 운영이 가능하다.

정답 1번

출제 포인트

커널의 주요 기능과 커널 구조의 특징을 구분하는 문제이다. 특히 프로세스 관리와 메모리 관리의 역할을 혼동하지 않아야 한다.

핵심 개념

프로세스 관리는 프로세스의 생성·종료, CPU 스케줄링, 동기화와 통신 등을 담당한다. 메모리 관리는 주기억장치 공간의 할당·회수와 가상 메모리 제공을 담당한다. 커널 구조는 대표적으로 단일형, 계층형, 마이크로커널 구조로 구분한다.

선지별 해설

① X — CPU를 프로세스에 배분하는 것은 프로세스 관리의 역할이지만, 작업 공간을 배치하고 실제 메모리보다 큰 가상 공간을 제공하는 것은 메모리 관리의 역할이다. 틀린 표현은 이 기능들을 모두 '프로세스 관리'에 포함한 부분이며, 올바르게는 '프로세스 관리는 CPU 배분 등을 담당하고, 메모리 관리는 메모리 공간의 할당과 가상 메모리를 담당한다'이다.
② O — 커널의 대표적인 구조는 단일형 구조, 계층형 구조, 마이크로커널 구조로 구분할 수 있다.
③ O — 운영체제에서 커널은 핵심 자원을 관리하고, 인터페이스는 사용자 및 응용 프로그램과 운영체제 사이에서 명령과 실행 결과를 전달한다.
④ O — 단일형 구조 커널은 기능 모듈이 커널 내부에 밀접하게 결합되어 있어 모듈 사이의 통신 오버헤드가 작고 실행 효율이 높다.

오답 포인트

①은 'CPU 배분'이라는 맞는 설명 뒤에 메모리 공간 배치와 가상 메모리 제공을 함께 붙여 모두 프로세스 관리의 기능처럼 보이게 한 선지이다.

NoteBase

다시 나오면 이렇게 판단

CPU의 배분·프로세스의 실행 순서는 프로세스 관리로, 메모리 공간의 할당·회수와 가상 메모리는 메모리 관리로 구분한다.

암기 포인트

프로세스 관리는 CPU와 실행, 메모리 관리는 공간과 가상 메모리이다.

NoteBase

문 17. NoSQL은 어떤 데이터 모델로 데이터를 저장하는지에 따라 분류가 가능하다. 다음 설명 중 가장 옳지 않은 것은?

① 컬럼 기반 데이터베이스는 컬럼 패밀리와 키의 쌍으로 데이터를 저장하는 형태를 의미한다.

② 그래프 기반 데이터베이스는 연관 데이터를 추천해 주거나 소셜 네트워크에서 친구 찾기 질의를 효율적으로 수행하는데 적합한 데이터베이스이다.

③ 문서 기반 데이터베이스는 컬럼 기반 데이터 모델을 확장한 형태로 JSON, XML 등과 같은 반정형 데이터를 저장한다.

④ 키-값 데이터베이스는 키와 값의 쌍으로 데이터를 저장하는 구조를 갖는다.

정답 3번

출제 포인트

NoSQL 데이터베이스를 저장 방식에 따라 키-값형, 문서형, 컬럼 패밀리형, 그래프형으로 구분하고 각 모델의 핵심 구조와 활용 분야를 판별하는 문제이다.

핵심 개념

키-값형은 고유한 키에 하나의 값을 대응시켜 저장한다. 문서형은 JSON이나 XML 같은 문서 단위로 반정형 데이터를 저장한다. 컬럼 패밀리형은 행 키를 기준으로 관련 컬럼을 컬럼 패밀리로 묶어 저장한다. 그래프형은 노드와 간선으로 개체와 관계를 표현한다.
핵심 구분은 키-값형은 키와 값, 문서형은 문서, 컬럼 패밀리형은 행 키와 컬럼 패밀리, 그래프형은 노드와 간선이다.

선지별 해설

① O — 컬럼 기반 데이터베이스는 키로 행을 식별하고 관련 컬럼을 컬럼 패밀리로 묶어 저장하는 모델이므로, 키와 컬럼 패밀리를 중심으로 데이터를 저장한다는 설명은 옳다.
② O — 그래프 기반 데이터베이스는 노드 사이의 관계를 간선으로 직접 표현하므로 추천 관계 탐색이나 소셜 네트워크의 친구 찾기처럼 연결 관계를 따라가는 질의에 적합하다.
③ X — 틀린 표현은 '컬럼 기반 데이터 모델을 확장한 형태'이다. 문서 기반 데이터베이스는 컬럼 기반 모델의 확장형이 아니라 독립적인 NoSQL 데이터 모델이며, JSON이나 XML 같은 문서 단위로 반정형 데이터를 저장한다.
④ O — 키-값 데이터베이스는 각 키에 값을 대응시키는 키와 값의 쌍으로 데이터를 저장하므로 옳다.

NoteBase

오답 포인트

③은 문서형이 JSON·XML 같은 반정형 데이터를 저장한다는 맞는 설명 뒤에 '컬럼 기반 데이터 모델을 확장한 형태'라는 틀린 관계를 붙였다. 저장 형식이 유연하다는 공통점만으로 문서형을 컬럼 패밀리형의 확장으로 보면 안 된다.

다시 나오면 이렇게 판단

각 모델을 저장 단위로 구별한다. 키와 값이면 키-값형, JSON·XML 문서이면 문서형, 행 키와 컬럼 패밀리이면 컬럼 패밀리형, 노드와 간선의 관계이면 그래프형으로 판단한다.

암기 포인트

키-값은 쌍, 문서는 JSON·XML, 컬럼은 패밀리, 그래프는 노드·간선이다.

NoteBase

문 18. 소프트웨어 아키텍처는 개발할 소프트웨어 구조, 주요 구성요소, 구성 요소간의 관계 및 상호작용을 결정하는 것을 의미한다. 아키텍처 스타일은 기능 분할과 배치 방법 그리고 제어 관계로 구분할 수 있다. 다음 중 기능 분할과 배치 방법에 해당하지 않는 것은?

① 데이터 흐름 스타일

② MVC 스타일

③ 클라이언트-서버 스타일

④ 계층 스타일

정답 1번

출제 포인트

아키텍처 스타일을 기능의 분할·배치 방법과 구성요소 사이의 제어 관계로 구분할 수 있는지를 묻는 문제이다.

핵심 개념

기능 분할과 배치 방법은 시스템의 책임을 어떤 단위로 나누고 어디에 배치하는지를 나타낸다. MVC 스타일, 클라이언트-서버 스타일, 계층 스타일이 이에 해당한다.
제어 관계는 구성요소 사이에서 처리와 데이터가 어떤 흐름으로 전달되는지를 나타낸다. 이 문항의 분류에서 데이터 흐름 스타일은 제어 관계에 해당한다.

선지별 해설

① X — 데이터 흐름 스타일은 데이터가 처리 단계를 따라 이동하면서 제어가 이어지는 방식이므로 기능 분할과 배치 방법에 해당하지 않는다. 올바르게는 제어 관계에 따른 아키텍처 스타일이다. 따라서 문제에서 찾는 선지이다.
② O — MVC 스타일은 소프트웨어의 기능과 책임을 모델, 뷰, 컨트롤러로 분리하여 배치하는 방식이다.
③ O — 클라이언트-서버 스타일은 요청 기능을 담당하는 클라이언트와 서비스를 제공하는 서버로 역할을 나누어 배치하는 방식이다.
④ O — 계층 스타일은 기능을 추상화 수준이나 책임에 따라 여러 계층으로 나누고 각 계층에 배치하는 방식이다.

오답 포인트

데이터 흐름 스타일도 구성요소를 설명하므로 기능 분할 방식처럼 보이기 쉽다. 그러나 이 문항에서는 데이터가 처리 단계를 따라 전달되는 제어 흐름에 초점을 둔다는 점이 핵심이다.

다시 나오면 이렇게 판단

역할을 무엇으로 나누고 어디에 놓는지를 설명하면 기능 분할·배치로, 처리나 데이터가 어떤 순서와 방향으로 전달되는지를 설명하면 제어 관계로 판단한다.

NoteBase

암기 포인트

MVC·클라이언트-서버·계층은 기능 분할과 배치, 데이터 흐름은 제어 관계이다.

NoteBase

문 19. 다음 중 다중 사용자 환경에서 CPU 시간을 일정 간격으로 균등하게 분할한 타임 슬라이스를 각 사용자에게 순차적으로 할당함으로써 마치 각자가 독점적으로 시스템을 사용하는 것처럼 느끼게 하는 운영체제 기법은?

① 오버레이(overlay)

② 시분할 시스템

③ 분산 처리 시스템

④ 실시간 처리 시스템

정답 2번

출제 포인트

CPU 시간을 짧은 시간 단위인 타임 슬라이스로 나누어 여러 사용자에게 순차적으로 할당하는 운영체제 기법을 구별하는 문제이다.

핵심 개념

시분할 시스템은 CPU 시간을 여러 개의 타임 슬라이스로 나누고 각 사용자나 프로세스에 번갈아 할당하여 대화식으로 처리하는 방식이다. 전환이 매우 빠르므로 각 사용자는 시스템을 독점하는 것처럼 느낀다.
오버레이는 프로그램 전체를 한꺼번에 메모리에 적재하기 어려울 때 필요한 부분만 같은 메모리 영역에 번갈아 올리는 기법이다.
분산 처리 시스템은 네트워크로 연결된 여러 컴퓨터가 작업과 자원을 나누어 처리하는 방식이다.
실시간 처리 시스템은 정해진 제한 시간 안에 반드시 결과를 내도록 응답 시간을 보장하는 방식이다.

선지별 해설

① X — 오버레이는 CPU 시간을 사용자별로 나누는 기법이 아니라, 필요한 프로그램 부분만 같은 메모리 영역에 번갈아 적재하는 메모리 관리 기법이다. 틀린 표현인 '오버레이'는 '시분할 시스템'으로 고쳐야 한다.
② O — 시분할 시스템은 CPU 시간을 타임 슬라이스로 균등하게 나누어 여러 사용자나 프로세스에 순차적으로 할당한다. 빠른 문맥 교환 때문에 각 사용자가 시스템을 독점하는 듯한 사용 환경을 제공한다.
③ X — 분산 처리 시스템은 하나의 CPU 시간을 여러 사용자에게 나누는 방식이 아니라, 네트워크로 연결된 여러 컴퓨터가 작업을 분담하는 방식이다. 틀린 표현인 '분산 처리 시스템'은 '시분할 시스템'으로 고쳐야 한다.
④ X — 실시간 처리 시스템의 핵심은 CPU 시간의 균등 분할이 아니라 정해진 제한 시간 안에 처리를 완료하는 것이다. 틀린 표현인 '실시간 처리 시스템'은 '시분할 시스템'으로 고쳐야 한다.

오답 포인트

'시간을 나눈다'는 표현 때문에 실시간 처리와 혼동하기 쉽다. 타임 슬라이스를 여러 사용자에게 순차 할당하면 시분할이고, 제한 시간 내 응답을 보장하면 실시간 처리이다.

NoteBase

다시 나오면 이렇게 판단

문장에서 ‘다중 사용자’, ‘타임 슬라이스’, ‘순차 할당’, ‘독점적으로 사용하는 느낌’을 확인하면 시분할 시스템으로 판단한다. ‘여러 컴퓨터의 협력’은 분산 처리, ‘제한 시간 내 응답’은 실시간 처리, ‘필요한 프로그램 부분만 메모리에 적재’는 오버레이로 구별한다.

암기 포인트

시분할은 한 CPU의 시간을 나누고, 분산 처리는 여러 컴퓨터가 일을 나누며, 실시간 처리는 마감 시간을 지킨다.

NoteBase

문 20. 관계 데이터 모델의 제약조건에 관한 다음 설명 중 가장 옳지 않은 것은?

① 기본키는 널(null) 값을 가질 수 있다.

② 데이터베이스의 상태가 변할 때 항상 무결성은 지켜져야 한다.

③ 데이터베이스의 일관성이 손상되지 않도록 보장해 주는 수단이다.

④ 외래키는 참조할 수 없는 값을 가질 수 없다는 규칙은 참조 무결성 제약조건이다.

정답 1번

출제 포인트

관계 데이터 모델의 무결성 제약조건 중 기본키의 개체 무결성과 외래키의 참조 무결성을 구별하는 문제이다.

핵심 개념

무결성 제약조건은 데이터베이스의 상태가 바뀌어도 데이터의 정확성과 일관성이 유지되도록 제한하는 규칙이다.
개체 무결성: 기본키의 각 속성은 널 값을 가질 수 없으며, 기본키 값은 각 튜플을 유일하게 식별한다.
참조 무결성: 외래키의 값은 참조되는 릴레이션의 기본키 값과 일치하거나 널 값이어야 한다. 다만 해당 외래키 속성에 별도의 NOT NULL 제약이 있으면 널 값도 허용되지 않는다.

선지별 해설

① X — '기본키는 널 값을 가질 수 있다'는 부분이 틀렸다. 기본키를 구성하는 속성은 개체 무결성 제약조건에 따라 널 값을 가질 수 없다. 올바르게는 '기본키는 널 값을 가질 수 없다'이다.
② O — 삽입·삭제·갱신 등으로 데이터베이스의 상태가 변하더라도 정의된 무결성 제약조건은 계속 만족되어야 한다.
③ O — 무결성 제약조건은 부적절한 데이터의 저장이나 변경을 제한하여 데이터베이스의 일관성이 손상되지 않도록 하는 수단이다.
④ O — 참조 무결성은 외래키에 참조 대상 릴레이션에 존재하지 않는 값을 저장하지 못하게 한다. 외래키는 참조되는 기본키 값과 일치하거나, 허용되는 경우 널 값이어야 한다.

오답 포인트

기본키와 외래키의 널 허용 여부를 같다고 생각하기 쉽다. 기본키는 반드시 널이 아니어야 하지만, 외래키는 참조되는 기본키 값과 일치하거나 제약조건이 허용하는 경우 널일 수 있다.

다시 나오면 이렇게 판단

기본키가 나오면 '유일하며 널이 아님'을, 외래키가 나오면 '참조되는 키와 일치하거나 널'을 기준으로 판단한다. 데이터 변경 전후에 이 규칙들이 계속 지켜져야 무결성이 유지된다.

NoteBase

암기 포인트

기본키는 유일·NOT NULL, 외래키는 참조값 일치 또는 NULL이다.

NoteBase

문 21. 소프트웨어 개발 모델 중 폭포수(Waterfall) 모델과 애자일(Agile) 모델에 관한 다음 설명 중 가장 옳지 않은 것은?

① 폭포수 모델은 상위 단계로 쉽게 회귀하며, 변경에 유연하다.

② 폭포수 모델은 전체 기능을 사전에 확정하고, 각 단계를 순차적으로 진행한다.

③ 애자일 모델은 짧은 개발 주기를 통해 빠르게 피드백을 받는다.

④ 애자일 모델은 변화하는 요구사항을 수용하며, 반복적이고 점진적으로 개발한다.

정답 1번

출제 포인트

폭포수 모델과 애자일 모델을 개발 단계의 진행 방식과 요구사항 변경 대응 방식으로 구별하는 문제이다.

핵심 개념

폭포수 모델은 요구사항 분석, 설계, 구현, 시험 등의 단계를 계획에 따라 순차적으로 수행하는 개발 모델이다. 앞 단계의 결과가 다음 단계의 기초가 되므로 개발 도중 이전 단계로 돌아가거나 요구사항을 바꾸기 어렵다.
애자일 모델은 짧은 주기로 작동 가능한 결과물을 반복적으로 만들고 피드백을 반영하여 점진적으로 개선하는 개발 모델이다. 변화하는 요구사항에 유연하게 대응하는 데 중점을 둔다.

선지별 해설

① X — 폭포수 모델은 단계가 순차적으로 연결되므로 상위 단계로 쉽게 회귀하지 못하고 개발 중 변경에도 유연하지 않다. 틀린 표현은 '상위 단계로 쉽게 회귀하며, 변경에 유연하다'이며, 올바르게는 '상위 단계로 회귀하기 어렵고 변경 대응이 경직되어 있다'이다.
② O — 폭포수 모델은 개발 초기에 전체 요구사항과 기능을 가능한 한 명확히 확정하고, 분석·설계·구현·시험 등의 단계를 순차적으로 진행한다.
③ O — 애자일 모델은 짧은 반복 개발 주기마다 결과물을 확인하고 사용자나 이해관계자의 피드백을 빠르게 반영한다.
④ O — 애자일 모델은 요구사항의 변화를 수용하면서 반복적·점진적으로 기능을 개발하고 개선한다.

오답 포인트

①의 '쉽게 회귀'와 '변경에 유연하다'는 폭포수 모델이 아니라 애자일 모델에 가까운 특성이다. 폭포수 모델의 단계 구분이 명확하다는 사실을 단계 간 이동도 자유롭다는 뜻으로 혼동하면 안 된다.

다시 나오면 이렇게 판단

'사전 확정·순차 진행·변경 곤란'이면 폭포수 모델로, '짧은 주기·반복·피드백·변경 수용'이면 애자일 모델로 판단한다.

NoteBase

암기 포인트

폭포수는 순차·고정, 애자일은 반복·변화 대응이다.

NoteBase

문 22. 다음 설명 중 가장 옳지 않은 것은?

① OSI 7의 5계층은 세션 계층으로 응용사이의 연결을 설정·관리·해제하는 통신제어 구조를 제공한다.

② OSI 7의 1계층은 데이터가 전송되는 동안 연결을 유지하거나 해제하는 기계적, 전기적, 기능적, 절차적 특성을 정의한다.

③ OSI 7의 3계층은 단말기 사이에 오류수정과 흐름제어를 수행하여 신뢰성 있고 명확한 데이터를 전송하는 계층이다.

④ OSI 7 계층은 상위계층과 하위계층으로 구분할 수 있다.

정답 3번

출제 포인트

OSI 7계층에서 각 계층의 번호와 핵심 기능을 정확히 연결하는 문제이다. 특히 3계층인 네트워크 계층과 오류·흐름 제어를 담당하는 계층을 구별해야 한다.

핵심 개념

OSI 7계층은 아래에서부터 물리, 데이터 링크, 네트워크, 전송, 세션, 표현, 응용 계층으로 구성된다.
물리 계층은 전송 매체와 인터페이스의 기계적·전기적·기능적·절차적 특성을 정한다. 데이터 링크 계층은 인접한 노드 사이의 프레임 전송과 오류·흐름 제어를 담당한다. 네트워크 계층은 논리 주소와 경로 선택을 이용해 패킷을 목적지까지 전달한다. 전송 계층은 종단 간 신뢰성 있는 전송을 담당하고, 세션 계층은 응용 프로세스 사이의 세션을 설정·관리·해제한다.

선지별 해설

① O — 5계층은 세션 계층이다. 응용 프로세스 사이의 통신 세션을 설정하고 관리하며 해제하는 기능을 제공한다.
② O — 1계층은 물리 계층이다. 통신 매체와 장치 인터페이스의 기계적·전기적·기능적·절차적 특성을 정의하고 물리 연결의 활성화·유지·해제를 다룬다.
③ X — 틀린 부분은 '3계층이 단말기 사이의 오류 수정과 흐름 제어를 수행한다'는 표현이다. 3계층은 네트워크 계층으로, 논리 주소 지정과 경로 선택을 통해 패킷을 목적지까지 전달한다. 인접 노드 사이의 오류·흐름 제어는 주로 2계층인 데이터 링크 계층이 담당하며, 종단 간 신뢰성 제어는 4계층인 전송 계층의 기능이다.
④ O — OSI 7계층은 일반적으로 1~4계층을 하위 계층, 5~7계층을 상위 계층으로 구분할 수 있다. 하위 계층은 데이터 전달, 상위 계층은 응용 서비스 지원에 중심을 둔다.

오답 포인트

③은 '오류 수정과 흐름 제어'라는 익숙한 기능을 네트워크 계층에 붙여 놓은 것이 함정이다. 네트워크 계층의 대표 기능은 오류 제어가 아니라 논리 주소 지정과 라우팅이다.

NoteBase

다시 나오면 이렇게 판단

계층 번호를 기능의 대표 키워드와 연결한다. 1계층은 신호·매체, 2계층은 프레임·인접 노드 오류 제어, 3계층은 패킷·논리 주소·라우팅, 4계층은 종단 간 전송, 5계층은 세션 관리로 판단한다.

암기 포인트

3계층은 네트워크, 핵심은 논리 주소와 라우팅이다. 오류·흐름 제어를 3계층의 대표 기능으로 연결하면 틀린다.

NoteBase

문 23. 다음 중 하나의 컴퓨터 시스템 하드웨어를 추상화하여 여러 개의 실행 환경(execution environment)이 존재하는 것으로 보이게 하고, 각 실행 환경이 마치 하나의 컴퓨터 시스템인 것으로 보이게 하는 기술에 해당하는 것은?

① 병렬처리(parallel processing) 기술

② 가상화(virtualization) 기술

③ 클라우드 컴퓨팅(cloud computing) 기술

④ 분산 컴퓨팅(distributed computing) 기술

정답 2번

출제 포인트

하나의 물리적 컴퓨터 하드웨어를 추상화하여 여러 개의 독립적인 실행 환경처럼 보이게 하는 기술이 무엇인지 묻는 문제이다.

핵심 개념

가상화는 물리적 컴퓨터의 CPU, 메모리, 저장장치 등의 자원을 추상화하여 여러 가상 실행 환경이 각각 독립된 컴퓨터처럼 동작하게 하는 기술이다.
병렬처리는 여러 처리 장치가 작업을 동시에 수행하는 기술이고, 클라우드 컴퓨팅은 네트워크를 통해 컴퓨팅 자원을 필요에 따라 제공하는 방식이며, 분산 컴퓨팅은 여러 독립된 컴퓨터가 통신하며 하나의 작업을 나누어 처리하는 방식이다.

선지별 해설

① X — 병렬처리 기술은 하나의 작업을 여러 프로세서나 코어가 동시에 처리하여 성능을 높이는 기술이다. 틀린 표현은 '병렬처리'이며, 제시된 설명에 맞는 표현은 '가상화'이다.
② O — 가상화 기술은 하나의 물리적 하드웨어를 추상화하여 여러 실행 환경이 각각 하나의 컴퓨터 시스템처럼 보이고 동작하게 한다.
③ X — 클라우드 컴퓨팅 기술은 네트워크를 통해 서버, 저장공간, 소프트웨어 등의 자원을 필요에 따라 제공하는 기술이다. 틀린 표현은 '클라우드 컴퓨팅'이며, 하나의 하드웨어를 여러 실행 환경으로 보이게 하는 기술은 '가상화'이다. 클라우드가 가상화를 활용할 수는 있지만 두 개념은 동일하지 않다.
④ X — 분산 컴퓨팅 기술은 네트워크로 연결된 여러 독립 컴퓨터가 작업을 분담하고 협력하여 처리하는 기술이다. 틀린 표현은 '분산 컴퓨팅'이며, 하나의 컴퓨터 하드웨어 안에 여러 실행 환경을 만드는 기술은 '가상화'이다.

NoteBase

오답 포인트

‘여러 개의 실행 환경’이라는 말만 보고 여러 처리 장치나 여러 컴퓨터를 사용하는 병렬처리·분산 컴퓨팅과 혼동하기 쉽다. 이 문항의 결정적 표현은 ‘하나의 컴퓨터 시스템 하드웨어를 추상화’한다는 부분이다.

다시 나오면 이렇게 판단

하나의 물리 장비를 여러 독립 환경처럼 나누면 가상화, 여러 처리 장치가 동시에 계산하면 병렬처리, 여러 독립 컴퓨터가 협력하면 분산 컴퓨팅, 네트워크로 자원을 서비스하면 클라우드 컴퓨팅으로 판단한다.

암기 포인트

한 대를 여러 대처럼 보이게 하면 가상화이다.

NoteBase

문 24. 다음 C프로그램의 MAX 값으로 옳은 것은?

```
#include <stdio.h>
int find_Max(int *a, int len)
{
    int maxval = 0;
    for(int i = 0; i < len; i = i + 2)
    {
        if (maxval < a[i])
            maxval = a[i];
    }
    return maxval;
}

void main()
{
    int arr[10] = {2, 88, 3, 23, 30, 17, 43, 77, 26, 65};
    int max = find_Max(arr, 10);
    printf("MAX = %d \n", max);
}
```

① 88

② 30

③ 43

④ 77

정답 3번

출제 포인트

반복문의 증감식에 따라 실제로 검사되는 배열 원소를 추적하고, 그중 최댓값을 구하는 문제이다.

조건 정리

반복 변수 *i*는 0에서 시작해 매번 2씩 증가한다. 따라서 검사하는 인덱스는 0, 2, 4, 6, 8이고, 해당 원소는 차례로 2, 3, 30, 43, 26이다. *maxval*의 초깃값은 0이며, 현재 원소가 *maxval*보다 클 때만 갱신된다.

NoteBase

계산 과정

$i = 0$: 2를 검사하여 $maxval = 2$가 된다.
$i = 2$: 3을 검사하여 $maxval = 3$이 된다.
$i = 4$: 30을 검사하여 $maxval = 30$이 된다.
$i = 6$: 43을 검사하여 $maxval = 43$이 된다.
$i = 8$: 26은 43보다 작으므로 갱신하지 않는다.
따라서 출력되는 MAX 값은 43이다.

선지 판단

① X — 88은 배열 전체의 최댓값이지만 인덱스 1에 있으므로 반복문이 검사하지 않는다. 올바르게는 검사 대상 원소의 최댓값인 43이다.
② X — 30은 $i = 4$까지 처리했을 때의 중간 최댓값일 뿐이며, 이후 인덱스 6의 43으로 갱신된다. 올바르게는 43이다.
③ O — 검사 대상인 인덱스 0, 2, 4, 6, 8의 원소 중 최댓값이 43이므로 맞다.
④ X — 77은 인덱스 7에 있어 반복문이 검사하지 않는다. 올바르게는 짝수 인덱스 원소만 비교하여 43이 된다.

오답 포인트

배열 전체를 검사한다고 생각하면 88을 고르기 쉽다. 핵심 함정은 증감식 $i = i + 2$이며, 이 때문에 홀수 인덱스의 88과 77은 비교 대상에서 제외된다.

다시 나오면 이렇게 판단

반복문의 초깃값, 계속 조건, 증감식을 먼저 확인해 실제 인덱스 열을 적고, 그 인덱스의 원소만 순서대로 대입하여 변수의 변화를 추적한다.

암기 포인트

$i = i + 2$이면 시작 인덱스와 같은 짝홀성을 가진 인덱스만 방문한다.

NoteBase

문 25. 관계형 데이터베이스에서의 키(key)에 관한 다음 설명 중 가장 옳지 않은 것은?

① 기본키는 한 릴레이션에 1개만 정의된다.

② 후보키는 유일성과 최소성을 만족하는 키이다.

③ 다른 릴레이션의 기본키를 속성으로 갖는 키를 후보키라고 한다.

④ 후보키 중에서 각 개체를 구별하기 위해 대표로 지정된 키를 기본키라고 한다.

정답 3번

출제 포인트

관계형 데이터베이스에서 후보키, 기본키, 외래키를 구별하는 문제이다.

핵심 개념

후보키는 튜플을 유일하게 식별하면서 불필요한 속성이 없는 최소 키이다. 기본키는 후보키 중 대표로 선택한 하나이며, 외래키는 다른 릴레이션 또는 같은 릴레이션의 키를 참조하여 릴레이션 사이의 관계를 나타내는 속성 또는 속성 집합이다.

선지별 해설

① O — 한 릴레이션에는 후보키가 여러 개 있을 수 있지만, 그중 대표로 선택하는 기본키는 하나만 정의된다. 다만 기본키 하나가 여러 속성으로 이루어진 복합키일 수는 있다.
② O — 후보키는 각 튜플을 서로 다르게 식별하는 유일성과, 그 구성에서 어느 속성도 더 제거할 수 없는 최소성을 모두 만족한다.
③ X — 틀린 표현은 '후보키'이다. 다른 릴레이션의 기본키 등을 참조하는 속성 또는 속성 집합은 외래키라고 한다. 후보키는 해당 릴레이션의 튜플을 유일성과 최소성을 갖추어 식별하는 키이다.
④ O — 여러 후보키 가운데 각 튜플을 대표적으로 식별하도록 선택한 키가 기본키이다.

오답 포인트

③은 다른 릴레이션의 키와 연결된다는 설명까지는 외래키의 특징인데, 마지막 명칭을 후보키로 바꾸어 놓았다. 후보키는 자기 릴레이션의 튜플 식별, 외래키는 릴레이션 간 참조라는 차이가 핵심이다.

다시 나오면 이렇게 판단

유일성과 최소성이 나오면 후보키, 후보키 중 대표 선택이 나오면 기본키, 다른 키를 참조한다는 표현이 나오면 외래키로 판단한다.

암기 포인트

후보키는 유일성·최소성, 기본키는 대표 후보키, 외래키는 참조 키이다.

NoteBase