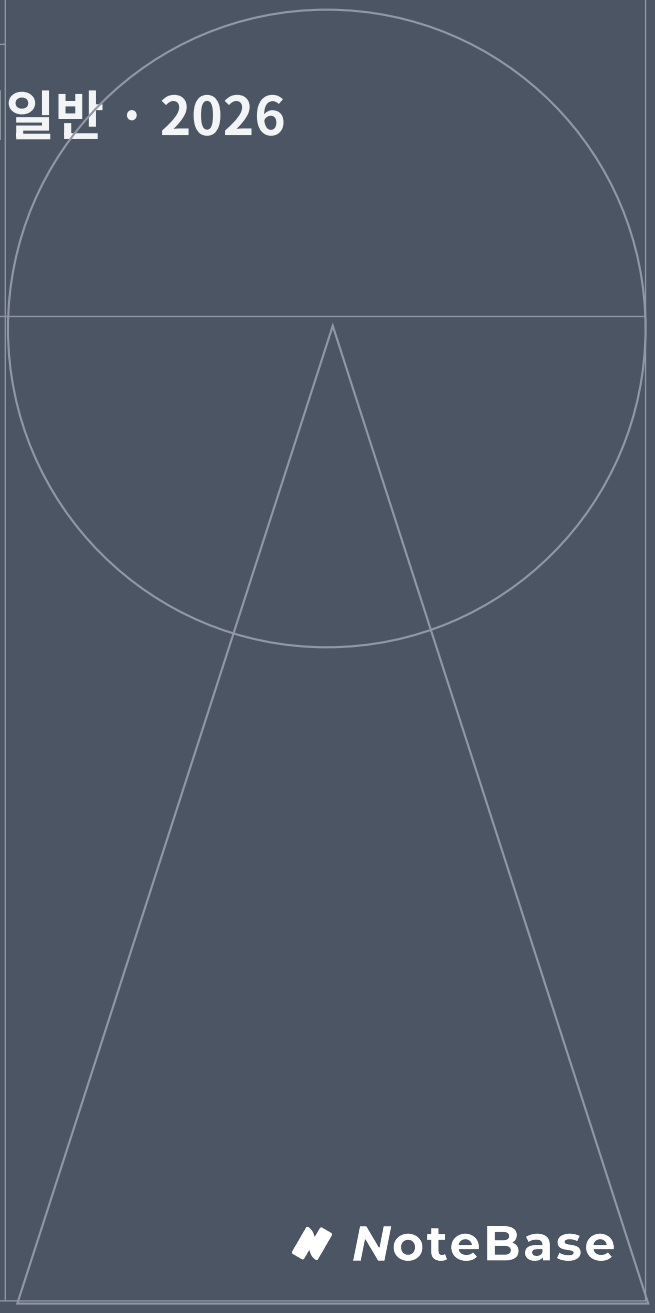
기출OX 해설집

공무원/법원직/9급 · 컴퓨터일반 · 2026



NoteBase

기출OX 해설집

공무원/법원직/9급 · 컴퓨터일반 · 2026

공무원/법원직/9급 · 컴퓨터일반 · 2026

문 1. 다음 중 전 가상화(full virtualization) 기반의 가상머신 방식과 비교해서 컨테이너 방식이 가지는 장점으로 가장 적절한 것은?

① 게스트 OS를 포함하지 않고 호스트 OS 커널을 공유하므로 오버헤드가 적고 시작 속도가 빠르다.

② 하드웨어를 완전히 에뮬레이션하므로 보안 격리 수준이 가상 머신 방식보다 높다.

③ 서로 다른 커널 버전의 OS를 하나의 호스트 위에서 자유롭게 실행할 수 있다.

④ 하드웨어와 호스트 OS 사이에 하이퍼바이저(hypervisor)를 배치하여 자원을 효율적으로 분배한다.

정답 1번

출제 포인트

전 가상화 기반 가상머신과 컨테이너를 게스트 OS의 포함 여부, 커널 공유 여부, 실행 오버헤드 측면에서 구분하는 문제이다.

핵심 개념

가상머신은 하이퍼바이저가 가상 하드웨어를 제공하고 각 가상머신이 독립된 게스트 OS와 커널을 실행하는 방식이다.
컨테이너는 애플리케이션과 그 실행 환경을 격리하되 호스트 OS의 커널을 공유하는 방식이다. 따라서 컨테이너는 게스트 OS를 별도로 구동하지 않아 일반적으로 가상머신보다 가볍고 시작이 빠르다.

NoteBase

선지별 해설

① O — 컨테이너는 게스트 OS 전체를 포함하지 않고 호스트 OS 커널을 공유한다. 그만큼 메모리·저장 공간 등의 오버헤드가 작고 생성 및 시작 속도가 빠르므로 적절한 장점이다.
② X — 틀린 표현은 '하드웨어를 완전히 에뮬레이션하므로 보안 격리 수준이 가상 머신 방식보다 높다'이다. 하드웨어 가상화와 독립된 게스트 OS를 이용하는 것은 가상머신 방식의 특징이며, 컨테이너는 호스트 커널을 공유하므로 일반적으로 가상머신보다 격리 경계가 약하다.
③ X — 틀린 표현은 '서로 다른 커널 버전의 OS를 하나의 호스트 위에서 자유롭게 실행할 수 있다'이다. 이는 독립된 게스트 OS와 커널을 구동하는 가상머신의 장점이다. 컨테이너는 호스트 커널을 공유하므로 호스트와 다른 종류의 커널을 독립적으로 실행하는 방식이 아니다.
④ X — 틀린 표현은 '하드웨어와 호스트 OS 사이에 하이퍼바이저를 배치'한다는 부분이다. 하이퍼바이저가 가상머신에 자원을 할당하는 구조는 가상머신 방식의 설명이며, 컨테이너는 호스트 OS 커널의 기능을 이용해 프로세스와 자원을 격리한다.

오답 포인트

②와 ④는 가상화 기술 전반의 특징처럼 보이지만 실제로는 하이퍼바이저와 가상 하드웨어를 사용하는 가상머신의 특징이다. ③도 여러 운영체제를 실행할 수 있다는 표현 때문에 컨테이너의 이식성과 혼동하기 쉽지만, 컨테이너 이식성은 호스트 커널을 공유한다는 제약 안에서 성립한다.

다시 나오면 이렇게 판단

'게스트 OS 없음·호스트 커널 공유·가벼움·빠른 시작'이면 컨테이너로 판단한다. '하이퍼바이저·가상 하드웨어·독립된 게스트 OS 또는 커널'이면 가상머신으로 판단한다.

암기 포인트

컨테이너는 커널을 공유해 가볍고 빠르며, 가상머신은 게스트 OS까지 분리해 격리한다.

NoteBase

문 2. 수강 정보를 관리하기 위한 [수강] 테이블을 아래와 같이 설계하려고 한다. 다음 중 [수강] 테이블의 STUDENT_ID가 [학생] 테이블에 존재하는 학생만을 참조하도록 보장하기 위한 가장 적절한 모델링 방법은?

[학생] 테이블
STUDENT_ID, STUDENT_NAME

[수강] 테이블
ENROLLMENT_ID, STUDENT_ID, COURSE_ID

① [수강] 테이블에 STUDENT_ID 대신 학생의 이름을 직접 저장하여 중복 관리한다.

② [수강] 테이블의 STUDENT_ID를 [학생] 테이블과 관계없는 독립적인 후보키(Candidate Key)로 지정하여 운용한다.

③ [수강] 테이블의 STUDENT_ID를 [학생] 테이블을 참조하는 기본키(Primary Key)로 선언하여 두 테이블을 하나의 키로 통합한다.

④ [수강] 테이블의 STUDENT_ID를 [학생] 테이블의 기본키인 STUDENT_ID를 참조하는 외래키(Foreign Key)로 정의한다.

정답 4번

출제 포인트

이 문제는 외래키가 테이블 사이의 참조 무결성을 보장하는 원리를 묻는 문제이다.

핵심 개념

외래키는 한 테이블의 열이 다른 테이블의 기본키나 고유키 값을 참조하도록 설정한 키이다. [수강]의 STUDENT_ID를 [학생]의 STUDENT_ID에 연결하면, [학생]에 존재하지 않는 STUDENT_ID가 [수강]에 입력되는 것을 제한하여 참조 무결성을 유지한다.
기본키는 한 테이블의 각 행을 식별하고, 후보키는 기본키가 될 수 있는 유일성과 최소성을 갖춘 키이며, 외래키는 테이블 사이의 참조 관계를 표현한다.

NoteBase

선지별 해설

① X — 학생 이름을 직접 저장하면 이름이 중복되거나 변경될 수 있고, [학생] 테이블에 실제로 존재하는 학생인지도 보장하지 못한다. 틀린 표현은 '학생의 이름을 직접 저장하여 중복 관리한다'이며, 올바르게는 학생을 식별하는 STUDENT_ID를 외래키로 저장한다.
② X — 후보키는 해당 테이블 안에서 행을 유일하게 식별하기 위한 키이므로, [학생] 테이블과 무관한 독립 키로 지정해도 참조 무결성이 생기지 않는다. 틀린 표현은 '관계없는 독립적인 후보키'이며, 올바르게는 [학생]의 기본키를 참조하는 외래키로 지정한다.
③ X — [수강]의 STUDENT_ID를 기본키로만 선언하면 [학생] 테이블을 참조한다는 제약이 설정되지 않는다. 또한 한 학생이 여러 과목을 수강할 수 있으므로 STUDENT_ID 하나만 기본키로 두면 같은 학생의 여러 수강 행을 저장하기 어렵다. 틀린 표현은 '참조하는 기본키로 선언'이며, 올바르게는 STUDENT_ID를 외래키로 선언하고 수강 행의 식별에는 ENROLLMENT_ID 등의 기본키를 사용한다.
④ O — [수강].STUDENT_ID를 [학생]의 기본키 STUDENT_ID를 참조하는 외래키로 정의하면, 존재하는 학생의 식별자만 수강 정보에 저장되도록 참조 무결성을 보장할 수 있다.

오답 포인트

③은 기본키와 외래키의 역할을 섞어 놓은 선지이다. 열 이름이 같거나 기본키로 선언되었다는 사실만으로 테이블 사이의 참조 관계가 생기지는 않으며, 참조 대상과의 외래키 제약을 명시해야 한다.

다시 나오면 이렇게 판단

'다른 테이블에 존재하는 값만 허용한다'는 조건이 보이면 참조하는 쪽의 열을 외래키로 설정하는지를 확인한다. 행 자체의 유일한 식별은 기본키, 다른 테이블 값의 참조는 외래키로 구분한다.

암기 포인트

기본키는 행 식별, 외래키는 테이블 연결과 참조 무결성이다.

NoteBase

문 3. 입력 데이터의 크기가 5×5이고, 3×3 크기의 커널(필터)를 사용하여 패딩(Padding) 없이 스트라이드(Stride)를 1로 설정하여 합성곱(Convolution) 연산을 수행하였다. 다음 <보기> 중 이에 관한 설명으로 옳은 것을 모두 고른 것은?

<보기>
ㄱ. 출력되는 특징 맵(Feature Map)의 크기는 2×2이다.
ㄴ. 커널(필터)을 이동시키는 간격인 스트라이드(Stride) 값을 크게 할수록 출력 특징 맵의 크기는 작아진다.
ㄷ. 합성곱 전후의 크기를 동일하게 맞추기 위해 입력 데이터의 외곽에 특정 값을 채워 넣는 패딩 방법을 수행해야 한다.
ㄹ. 합성곱 연산은 커널(필터)의 각 요소와 입력 데이터의 대응 요소를 각각 곱한 9개 값의 평균을 계산하는 방식이다.

① ㄱ, ㄴ
② ㄴ, ㄷ
③ ㄷ, ㄹ
④ ㄱ, ㄹ

정답 2번

출제 포인트

합성곱의 출력 크기 공식을 적용하고, 스트라이드·패딩·합성곱 연산의 의미를 구분하는 문제이다.

조건 정리

입력 크기: 5×5
커널 크기: 3×3
패딩: $P = 0$
스트라이드: $S = 1$
구해야 할 값: 출력 특징 맵의 크기와 보기 ㄱ~ㄹ의 참·거짓

사용할 공식

한 변의 출력 크기는 다음과 같다.
$\left\lfloor \frac{N + 2P - F}{S} \right\rfloor + 1$
여기서 N은 입력 크기, P는 패딩 크기, F는 커널 크기, S는 스트라이드이다. 커널이 입력 위에 완전히 놓이는 위치의 개수를 구해야 하므로 이 공식을 사용한다.

NoteBase

계산 과정

가로와 세로에 각각 같은 공식을 적용한다.
$\left\lfloor \frac{5+2\times 0-3}{1} \right\rfloor + 1 = 3$
따라서 출력 특징 맵의 크기는 3 × 3이다. 입력과 같은 5 × 5 크기를 유지하려면 커널의 가장자리를 보완하도록 한쪽 기준 1칸의 패딩이 필요하다.

선지별 해설

① X — ‘ㄱ, ㄴ’ 중 ㄱ이 틀렸다. 패딩 없이 3 × 3 커널을 스트라이드 1로 적용하면 출력은 2 × 2가 아니라 3 × 3이다. 올바른 조합은 ‘ㄴ, ㄷ’이다.
② O — ㄴ과 ㄷ이 모두 옳다. 스트라이드가 커지면 커널을 놓는 위치 수가 줄어 출력 특징 맵이 작아진다. 또한 합성곱 전후의 크기를 같게 유지하려면 입력 외곽에 값을 채우는 패딩이 필요하다.
③ X — ‘ㄷ, ㄹ’ 중 ㄹ이 틀렸다. 일반적인 합성곱은 대응 원소들의 곱을 모두 더한 값을 출력하며, 9개 값의 평균을 구하는 연산이 아니다. 올바른 조합은 ‘ㄴ, ㄷ’이다.
④ X — ‘ㄱ, ㄹ’은 둘 다 틀렸다. 출력은 3 × 3이고, 합성곱은 원소별 곱의 합을 계산한다. 올바른 조합은 ‘ㄴ, ㄷ’이다.

오답 포인트

ㄱ에서는 출력 크기 공식의 마지막 ‘+1’을 빠뜨리면 2 × 2로 잘못 판단하기 쉽다. ㄹ에서는 합성곱의 ‘곱한 뒤 합산’을 평균 풀링의 ‘평균 계산’과 혼동하기 쉽다.

다시 나오면 이렇게 판단

입력·커널·패딩·스트라이드를 출력 크기 공식에 먼저 대입한 뒤, 합성곱은 원소별 곱의 합이고 평균 풀링은 평균이라는 차이를 확인한다.

암기 포인트

출력 크기는 ‘입력 + 양쪽 패딩 - 커널’을 스트라이드로 나눈 뒤 1을 더하며, 합성곱은 곱의 합이다.

NoteBase

문 4. 비어 있는 최소힙(min heap)에 다음과 같은 데이터를 차례로 삽입했을 때 루트 노드의 오른쪽 자식노드에 존재하는 값은 얼마인가?

15, 3, 27, 1

① 1
② 3
③ 15
④ 27

정답 4번

출제 포인트

이 문제는 최소힙에 원소를 차례로 삽입하면서 완전이진트리의 형태와 부모가 자식보다 작거나 같은 최소힙 성질을 유지하는 과정을 묻는 문제이다.

올바른 순서

1. 15 삽입 — 비어 있는 힙의 루트에 15가 들어간다: [15]
2. 3 삽입 — 다음 자리인 왼쪽 자식에 넣은 뒤 부모 15보다 작으므로 교환한다: [3, 15]
3. 27 삽입 — 루트의 오른쪽 자식 자리에 들어가며 부모 3보다 크므로 그대로 둔다: [3, 15, 27]
4. 1 삽입 — 다음 자리인 15의 왼쪽 자식에 넣고 15와 교환한 다음, 다시 3과 교환한다: [1, 3, 27, 15]

왜 이 순서인가: 최소힙 삽입은 완전이진트리의 다음 빈자리에 값을 넣은 뒤 부모와 비교하며 위로 올리므로, 1의 상향 교환은 기존 오른쪽 자식 27의 자리를 바꾸지 않는다. 최종적으로 루트의 오른쪽 자식은 27이다.

선지별 해설

① X — 1은 최종 힙의 루트 값이다. 틀린 값 1을 고친 루트의 오른쪽 자식 값은 27이다.
② X — 3은 최종 힙에서 루트의 왼쪽 자식 값이다. 틀린 값 3을 고친 루트의 오른쪽 자식 값은 27이다.
③ X — 15는 최종 힙에서 값 3의 왼쪽 자식인 리프 노드에 놓인다. 틀린 값 15를 고친 루트의 오른쪽 자식 값은 27이다.
④ O — 최종 배열 표현이 [1, 3, 27, 15]이므로 루트 1의 오른쪽 자식은 27이다.

오답 포인트

마지막에 삽입한 1이 위로 올라가는 과정에서 모든 노드의 위치가 바뀐다고 생각하기 쉽다. 1은 자신의 조상인 15와 3하고만 차례로 교환하므로 다른 가지에 있는 27은 루트의 오른쪽 자리에 그대로 남는다.

NoteBase

다시 나오면 이렇게 판단

최소힙 삽입 문제는 새 값을 완전이진트리의 다음 빈자리에 놓고, 부모보다 작은 동안에만 부모와 교환한다. 각 삽입이 끝날 때마다 배열 형태를 기록하면 원하는 부모·자식 위치를 정확히 찾을 수 있다.

암기 포인트

최소힙 삽입은 '끝에 넣고, 부모보다 작으면 위로'이며 교환은 새 노드의 조상 경로에서만 일어난다.

NoteBase

문 5. 교착상태(Deadlock)에 관한 다음 설명 중 가장 옳지 않은 것은?

① 점유하며 대기(Hold and wait) 조건은 최소 하나의 자원을 보유한 상태에서, 타인이 점유한 추가 자원을 얻기 위해 대기하는 상태를 말한다.

② 비선점(No preemption) 조건을 방지하는 방식은 자원 요청 시 즉시 할당이 불가능하다면, 보유 자원을 모두 해제하고 필요 자원 확보 시에만 재시작한다.

③ 은행원 알고리즘(Banker's Algorithm)은 최대 자원 요구량을 미리 선언하며, 시스템이 안전 상태를 유지할 수 있을 때만 자원을 배분한다.

④ 교착상태를 해소하기 위해 항상 가장 많은 자원을 점유한 프로세스를 먼저 종료하고, 종료된 프로세스에게 할당된 모든 자원을 회수한다.

정답 4번

출제 포인트

교착상태의 발생 조건과 예방·회피·회복 기법의 차이를 구별하는 문제이다.

핵심 개념

교착상태는 여러 프로세스가 서로 상대방이 가진 자원을 기다리면서 어느 것도 진행하지 못하는 상태이다. 발생에는 상호배제, 점유하며 대기, 비선점, 순환대기의 네 조건이 모두 필요하다. 예방은 이 조건 중 하나를 성립하지 않게 하고, 회피는 자원을 배분한 뒤에도 안전 상태가 유지되는지를 확인하며, 회복은 교착상태를 탐지한 뒤 프로세스 종료나 자원 선점으로 해소한다.

선지별 해설

① O — 점유하며 대기는 프로세스가 하나 이상의 자원을 보유한 채 다른 프로세스가 점유한 추가 자원을 기다리는 조건이다.
② O — 요청한 자원을 즉시 받을 수 없을 때 현재 보유 자원을 반납하게 하고, 기존 자원과 필요한 자원을 다시 확보한 뒤 재시작하게 하면 자원을 계속 빼앗을 수 없다는 비선점 조건을 깨뜨릴 수 있다.
③ O — 은행원 알고리즘은 각 프로세스의 최대 자원 요구량을 미리 알고, 자원 할당 후에도 모든 프로세스가 완료될 수 있는 안전 순서가 존재할 때만 자원을 배분하는 교착상태 회피 기법이다.
④ X — 틀린 표현은 '항상 가장 많은 자원을 점유한 프로세스를 먼저 종료'한다는 부분이다. 교착상태 회복에서 종료할 희생 프로세스는 우선순위, 수행 시간, 남은 시간, 보유·요구 자원, 종료 및 재시작 비용 등을 종합하여 선택하며, 자원 점유량만으로 항상 결정하지 않는다. 올바르게는 '회복 비용을 최소화하도록 희생 프로세스를 선택해 종료하고 그 자원을 회수할 수 있다'이다.

NoteBase

오답 포인트

④의 '가장 많은 자원'은 희생 프로세스 선택에 고려할 수 있는 요소이지만 절대적인 규칙은 아니다. '항상'이라는 단정이 회복 정책의 선택 기준을 하나로 고정하므로 틀린다.

다시 나오면 이렇게 판단

발생 조건을 무너뜨리면 예방, 안전 상태를 확인한 뒤 할당하면 회피, 발생을 탐지한 뒤 프로세스 종료·자원 선점으로 없애면 회복으로 판단한다. 회복 대상 선정에 '항상', '반드시'와 같은 단정이 붙으면 비용과 우선순위 등 복수 기준을 배제하는지 확인한다.

암기 포인트

예방은 조건 제거, 은행원 알고리즘은 안전 상태 회피, 종료·선점은 탐지 후 회복이다.

NoteBase

문 6. 다음 C 언어 프로그램의 실행결과로 옳은 것은?

```
#include <stdio.h>
int main()
{   char a[] = "KOREA";
    char *p = &a[1];
    *(p+3) = 'D';
    p++;
    printf("%s", p);
}
```

① READ

② KORED

③ RED

④ KOREA

정답 3번

출제 포인트

문자 배열에서 포인터가 가리키는 위치와 포인터 연산에 따른 문자 변경 및 문자열 출력 범위를 순서대로 추적하는 문제이다.

올바른 순서

1. char a[] = "KOREA" — 배열 a에는 K, O, R, E, A와 문자열 종료 문자 \0이 차례로 저장된다.
2. char *p = &a[1] — p는 두 번째 문자 O가 있는 a[1]을 가리킨다.
3. *(p+3) = 'D' — p+3은 a[4]이므로 마지막 문자 A가 D로 바뀌어 배열의 문자열은 KORED가 된다.
4. p++ — p는 다음 원소인 a[2], 즉 R을 가리킨다.
5. printf("%s", p) — p가 가리키는 R부터 문자열 종료 문자까지 출력하므로 결과는 RED이다.

앞의 대입으로 배열 내용이 바뀐 뒤 p가 한 칸 이동하므로, 전체 배열 KORED가 아니라 a[2]부터 시작하는 RED가 출력된다.

NoteBase

선지별 해설

① X — READ는 출력되지 않는다. p+3이 가리키는 곳은 a[4]이므로 A가 D로 바뀌고, p++ 후에는 a[2]부터 출력한다. 올바른 결과는 RED이다.
② X — KORED는 문자 변경 후 배열 전체를 a[0]부터 읽은 문자열이다. 그러나 출력 시 p는 a[2]를 가리키므로 올바른 결과는 RED이다.
③ O — p++ 후 p는 a[2]의 R을 가리키며, 그 위치부터 종료 문자까지의 문자열은 RED이다.
④ X — KOREA는 문자 대입과 포인터 이동을 모두 반영하지 않은 원래 문자열이다. A는 D로 변경되고 출력 시작점은 a[2]이므로 올바른 결과는 RED이다.

오답 포인트

② KORED는 배열의 변경만 반영하고 p++에 따른 출력 시작점 이동을 놓친 선택지이다. 또한 *(p+3)은 p를 이동시키는 연산이 아니라 p를 기준으로 세 칸 뒤의 원소 값을 바꾸는 연산이라는 점을 구분해야 한다.

다시 나오면 이렇게 판단

배열의 각 문자에 인덱스를 붙인 뒤, 대입문마다 배열 내용과 포인터 위치를 따로 갱신한다. %s는 포인터가 현재 가리키는 문자부터 문자열 종료 문자까지 출력한다.

암기 포인트

*(p+n)은 n칸 뒤의 값을 바꾸고, p++은 포인터 자체를 한 칸 옮긴다. %s는 현재 포인터 위치부터 출력한다.

NoteBase

문 7. 메모리 참조의 2가지 지역성(locality)에 관한 아래 설명에서 괄호 안에 들어갈 표현으로 가장 옳은 것은?

• 시간지역성(temporal locality): (가) 사용된 데이터가 가까운 미래에 다시 사용될 가능성이 높은 성질
• 공간지역성(spatial locality): 특정 위치의 데이터가 사용되었을 때, (나) 위치의 데이터가 가까운 미래에 사용될 가능성이 높은 성질

① (가): 최근에(recently), (나): 동일한(same)
② (가): 최근에(recently), (나): 인접한(nearby)
③ (가): 빈번히(frequently), (나): 인접한(nearby)
④ (가): 빈번히(frequently), (나): 동일한(same)

정답 2번

출제 포인트

메모리 참조의 시간지역성과 공간지역성을 각각 시간의 반복성과 주소의 인접성으로 구분하는 문제이다.

핵심 개념

시간지역성: 최근에 참조한 데이터가 가까운 미래에 다시 참조될 가능성이 높은 성질이다. 키워드는 최근·재사용이다.
공간지역성: 특정 위치의 데이터를 참조하면 그 위치와 인접한 주소의 데이터도 가까운 미래에 참조될 가능성이 높은 성질이다. 키워드는 인접 주소이다.
시간지역성은 같은 데이터를 다시 쓰는 시간적 반복에, 공간지역성은 가까운 위치의 데이터를 이어서 쓰는 주소적 근접성에 초점을 둔다.

NoteBase

선지별 해설

① X
(가)의 '최근에'는 시간지역성의 정의에 맞지만, (나)의 '동일한'은 틀린 표현이다. 공간지역성은 동일한 위치가 아니라 인접한 위치의 데이터가 참조될 가능성이 높은 성질이다.

② O
시간지역성은 최근에 사용된 데이터의 재사용 가능성이 높다는 성질이고, 공간지역성은 인접한 위치의 데이터가 사용될 가능성이 높다는 성질이므로 두 표현이 모두 옳다.

③ X
(나)의 '인접한'은 공간지역성의 정의에 맞지만, (가)의 '빈번히'는 틀린 표현이다. 시간지역성은 빈번히 사용된 데이터가 아니라 최근에 사용된 데이터가 가까운 미래에 다시 사용될 가능성이 높은 성질이다.

④ X
(가)의 '빈번히'와 (나)의 '동일한'이 모두 틀린 표현이다. 올바르게는 (가)는 '최근에', (나)는 '인접한'이다.

오답 포인트

'빈번히 사용됨'도 반복 사용을 떠올리게 하지만 시간지역성의 기준은 사용 횟수가 아니라 최근에 참조했는지이다. 또한 '동일한 위치'의 재참조는 시간지역성에 가깝고, 공간지역성은 인접한 위치의 참조를 뜻한다.

다시 나오면 이렇게 판단

시간지역성은 '방금 쓴 것을 다시 쓴다', 공간지역성은 '방금 쓴 곳의 주변을 이어서 쓴다'로 판단한다. 따라서 시간에는 최근, 공간에는 인접이라는 표현을 대응시킨다.

암기 포인트

시간은 최근 데이터의 재사용, 공간은 인접 데이터의 사용이다.

NoteBase

문 8. [도서] 테이블에서 출판사별 평균 가격이 20,000원 이상인 출판사의 출판사명과 평균 가격을 검색하고자 한다. 또한 결과는 평균 가격이 높은 순서대로 출력하려고 한다. 다음 SQL문에서 빈칸 ( )에 들어갈 내용으로 가장 적절한 것은?

```
SELECT 출판사명, AVG(가격) FROM 도서 (        ) ORDER BY AVG(가격) DESC;
```

① GROUP BY 가격 HAVING 출판사명 >= 20000

② HAVING AVG(가격) >= 20000 WHERE 출판사명

③ WHERE AVG(가격) >= 20000 GROUP BY 출판사명

④ GROUP BY 출판사명 HAVING AVG(가격) >= 20000

정답 4번

출제 포인트

출판사별로 행을 묶는 GROUP BY와 집계 결과인 평균 가격을 거르는 HAVING의 역할 및 작성 순서를 묻는 문제이다.

핵심 개념

GROUP BY는 같은 값을 가진 행을 그룹으로 묶는 절이며, 이 문항에서는 출판사별 평균을 구하므로 출판사명을 기준으로 묶는다.
HAVING은 GROUP BY로 만든 그룹의 집계 결과에 조건을 적용하는 절이므로 AVG(가격) >= 20000을 사용한다.
이 문장의 핵심 순서는 FROM 도서 → GROUP BY 출판사명 → HAVING AVG(가격) >= 20000 → ORDER BY AVG(가격) DESC이다.

NoteBase

선지별 해설

① X - GROUP BY 가격은 같은 가격끼리 묶으므로 출판사별 평균을 구할 수 없다. 또한 HAVING 출판사명 >= 20000은 문자 성격의 출판사명을 금액 20000과 비교한 잘못된 조건이다. 올바르게는 GROUP BY 출판사명 HAVING AVG(가격) >= 20000이다.
② X - HAVING이 GROUP BY 없이 놓였고, WHERE 출판사명은 비교 조건이 없는 불완전한 절이다. 올바르게는 GROUP BY 출판사명 뒤에 HAVING AVG(가격) >= 20000을 둔다.
③ X - WHERE는 그룹화 전의 개별 행을 거르므로 AVG(가격) 같은 집계 함수의 결과 조건을 둘 수 없다. 올바르게는 GROUP BY 출판사명 HAVING AVG(가격) >= 20000이다.
④ O - 출판사명으로 그룹화한 뒤 각 그룹의 평균 가격이 20000 이상인지 HAVING으로 거르므로 요구와 정확히 일치한다. 이어지는 ORDER BY AVG(가격) DESC가 평균 가격의 내림차순 정렬을 수행한다.

오답 포인트

집계 조건 AVG(가격) >= 20000을 WHERE에 쓰는 것이 핵심 함정이다. WHERE는 그룹화 전의 행 조건이고, HAVING은 그룹화 후의 집계 결과 조건이다.

다시 나오면 이렇게 판단

'무엇별로'에 해당하는 열은 GROUP BY에 두고, 평균·합계·개수 등 집계값에 대한 조건은 HAVING에 둔다. 결과 정렬은 마지막 ORDER BY에서 처리한다.

암기 포인트

행 조건은 WHERE, 그룹 기준은 GROUP BY, 집계 조건은 HAVING, 결과 정렬은 ORDER BY이다.

NoteBase

문 9. 다음과 같이 정의된 C 언어 함수들을 가정할 때, 아래의 for-반복문이 실행되는 동안 함수 decr()이 호출되는 횟수는 몇 번인가?

```
#include <stdio.h>
long min(long x, long y) { return x < y ? x : y; }
long max(long x, long y) { return x < y ? y : x; }
void decr(long *xp, long v) { *xp -= v; }
long sq(long x) { return x*x; }

void main() {
  long x = 20, y = 50, t = 0, i;
  for (i = max(x, y) - 1; i >= min(x, y); decr(&i, 1))
      t += sq(i);
}
```

① 20

② 30

③ 40

④ 50

정답 2번

출제 포인트

for문의 초기식·조건식·증감식 실행 순서를 바탕으로 증감식에 있는 decr()의 호출 횟수를 구하는 문제이다.

조건 정리

max(x, y)는 50이고 min(x, y)는 20이다. 따라서 i의 초깃값은 50 - 1 = 49이다. 조건식은 i가 20 이상인 동안 참이며, 반복 본문이 한 번 끝날 때마다 증감식 decr(&i, 1)이 호출되어 i가 1씩 감소한다.

사용할 공식

1씩 감소하면서 위끝값과 아래끝값을 모두 포함하는 반복 횟수는 위끝값-아래끝값+1이다. 이 문항에서는 i가 49부터 20까지 1씩 감소하므로 이 공식을 적용한다.

NoteBase

계산 과정

반복 본문이 실행될 때의 i는 49, 48, ..., 20이다. 이 정수들의 개수는 49 − 20 + 1 = 30이다. i가 20인 반복의 본문이 끝난 뒤에도 decr(&i, 1)이 한 번 호출되어 i가 19가 되고, 다음 조건 검사에서 반복이 종료된다. 따라서 decr()은 총 30번 호출된다.

선지 판단

① X — 20은 반복 조건의 하한값이지 호출 횟수가 아니다. '20번'이 아니라 올바르게는 '30번'이다.
② O — i가 49부터 20까지 변하는 30회의 반복마다 증감식의 decr()이 한 번씩 호출되므로 30번이다.
③ X — 반복 횟수 계산 결과는 40이 아니다. '40번'이 아니라 올바르게는 '30번'이다.
④ X — 50은 max(x, y)의 값이지 호출 횟수가 아니다. 또한 i는 50이 아니라 49에서 시작한다. '50번'이 아니라 올바르게는 '30번'이다.

오답 포인트

50과 20을 그대로 호출 횟수로 착각하거나, for문의 증감식이 반복 본문 뒤에 실행된다는 점을 놓치기 쉽다. 특히 마지막 본문이 i = 20에서 실행된 뒤에도 decr()이 호출된 다음 조건을 다시 검사한다.

다시 나오면 이렇게 판단

for문은 초기식 → 조건 검사 → 본문 → 증감식 → 조건 재검사의 순서로 추적한다. 증감식 속 함수의 호출 횟수는 정상적으로 완료된 반복 본문의 횟수와 같다.

암기 포인트

for문의 증감식은 각 반복 본문이 끝난 뒤 한 번 실행되며, 1씩 감소하는 범위에서 양 끝을 모두 포함한 횟수는 큰 값-작은 값+1로 센다.

NoteBase

문 10. 다음 C 언어 프로그램의 실행 결과로 옳은 것은?

```
#include <stdio.h>
int main() {
    int data[] = {10, 20, 30, 40, 50};
    int *ptr = data;
    ptr++;
    *ptr += 5;

    printf("%d", *(ptr+1) + *(ptr));

    return 0;
}
```

① 35
② 50
③ 55
④ 75

정답 3번

출제 포인트

배열을 가리키는 포인터가 이동한 뒤 역참조 연산으로 배열 원소를 변경하고, 포인터 산술로 참조한 두 값을 더할 수 있는지를 묻는 문제이다.

조건 정리

초기 배열은 data = {10, 20, 30, 40, 50}이다.
ptr = data이므로 처음에는 ptr이 data[0]을 가리킨다.
ptr++ 실행 후 ptr은 다음 원소인 data[1]을 가리킨다.
*ptr += 5는 ptr 자체가 아니라 ptr이 가리키는 data[1]의 값을 20에서 25로 변경한다.

NoteBase

계산 과정

1. ptr++ 이후 ptr은 data[1]을 가리킨다.
2. *ptr += 5 실행 후 배열은 {10, 25, 30, 40, 50}이 된다.
3. *(ptr+1)은 data[2]의 값 30이고, *(ptr)은 data[1]의 값 25이다.
4. printf가 출력하는 값은 30 + 25 = 55이다.

선지 판단

① X — 35는 계산 결과와 다르다. *ptr += 5는 data[1]을 25로 변경하므로 올바른 출력값은 35가 아니라 55이다.
② X — 50은 *ptr += 5에 따른 값 변경을 반영하지 않고 30 + 20으로 계산한 값이다. 올바르게는 변경된 25를 사용하여 30 + 25 = 55가 된다.
③ O — *(ptr+1) = 30이고 *(ptr) = 25이므로 합은 55이다.
④ X — 75는 ptr+1이 가리키는 원소를 잘못 잡은 값이다. ptr이 data[1]을 가리키므로 ptr+1은 data[2]을 가리키며, 올바른 출력값은 75가 아니라 55이다.

오답 포인트

ptr++이 포인터가 가리키는 위치를 data[0]에서 data[1]로 옮긴다는 점과, *ptr += 5가 포인터 주소가 아니라 해당 배열 원소의 값을 바꾼다는 점을 혼동하기 쉽다. 특히 ②는 값 변경을 누락했을 때 나오는 오답이다.

다시 나오면 이렇게 판단

포인터 문제는 각 문장을 실행할 때마다 포인터가 가리키는 배열 인덱스와 배열의 현재 값을 따로 기록한 뒤, 마지막 역참조 식을 배열 원소로 치환하여 계산한다.

암기 포인트

ptr++은 가리키는 원소를 한 칸 이동시키고, *ptr의 변경은 그 원소의 실제 값을 변경한다.

NoteBase

문 11. 클라우드 컴퓨팅에 관한 다음 설명 중 가장 옳지 않은 것은?

① IaaS(Infrastructure as a Service) 환경에서 소비자는 가상 서버와 네트워크 등 인프라 자원을 제공받으며, 해당 자원의 설정과 운영에 대한 직접적인 제어권과 관리 책임을 갖는다.

② PaaS(Platform as a Service)는 완성된 소프트웨어를 서비스로 제공하는 모델로, 소비자는 제공되는 기능만을 이용하며 전체적인 컴퓨팅 자원에 대해 매우 제한적인 관리 제어 권한을 갖는다.

③ 자원 풀링(Resource Pooling)은 물리 및 가상 자원을 하나의 풀로 관리하며, 소비자의 수요에 따라 자원을 동적으로 할당하는 방식으로, 일반적으로 통계적 다중화 방식을 기반으로 운영된다.

④ 가상화(Virtualization)는 물리 자원을 가상 자원으로 전환시키는 과정으로, 물리 서버는 가상 서버로, 물리 스토리지는 가상 디스크로 추상화되어 사용자에게 제공된다.

정답 2번

출제 포인트

클라우드 서비스 모델인 IaaS·PaaS·SaaS의 제공 범위와 자원 풀링·가상화의 의미를 구별하는 문제이다.

핵심 개념

IaaS는 서버·스토리지·네트워크 같은 가상화된 인프라를 제공하며, 소비자가 운영체제와 배포한 애플리케이션 등을 관리하는 모델이다.
PaaS는 애플리케이션의 개발·실행·배포에 필요한 플랫폼 환경을 제공하며, 소비자는 주로 자신이 배포한 애플리케이션을 관리한다.
SaaS는 제공자가 운영하는 완성된 소프트웨어를 소비자가 서비스로 이용하는 모델이다.
자원 풀링은 여러 소비자가 공유하는 자원 풀에서 수요에 따라 자원을 동적으로 배정하는 것이고, 가상화는 물리 자원을 논리적인 가상 자원으로 추상화하는 기술이다.

선지별 해설

① O — IaaS 소비자는 가상 서버와 네트워크 등의 인프라를 제공받고, 제공받은 가상 자원의 설정과 운영체제·애플리케이션 관리에 직접 관여한다. 다만 물리 장비와 기반 클라우드 인프라는 제공자가 관리한다.
② X — 틀린 표현은 ‘PaaS는 완성된 소프트웨어를 서비스로 제공한다’이다. 이는 SaaS의 설명이다. 올바르게는 PaaS가 애플리케이션을 개발·실행·배포할 수 있는 플랫폼을 제공하고, 소비자가 그 위에 배포한 애플리케이션을 관리하는 모델이다.
③ O — 자원 풀링은 물리·가상 자원을 공동 풀로 구성하고 여러 소비자의 수요에 맞춰 동적으로 할당하는 방식이다. 여러 이용자의 변동하는 수요를 묶어 처리하는 통계적 다중화와도 부합한다.
④ O — 가상화는 물리 서버·스토리지 등의 자원을 가상 서버·가상 디스크 같은 논리적 자원으로 추상화하여 제공하는 기술이다.

NoteBase

오답 포인트

②는 PaaS와 SaaS의 역할을 바꾸어 놓았다. ‘완성된 소프트웨어의 기능을 이용한다’는 표현은 SaaS이고, ‘애플리케이션을 개발·배포할 플랫폼을 이용한다’는 표현이 PaaS이다.

다시 나오면 이렇게 판단

서비스 제공 범위를 기준으로 구별한다. 인프라를 빌리면 IaaS, 개발·실행 플랫폼을 빌리면 PaaS, 완성된 소프트웨어를 이용하면 SaaS이다.

암기 포인트

IaaS는 인프라, PaaS는 개발 플랫폼, SaaS는 완성 소프트웨어이다.

NoteBase

문 12. 분산 시스템(distributed system)이 가지는 특징으로 다음 설명 중 가장 옳지 않은 것은?

① 여러 개의 단위 시스템들이 네트워크를 통해 상호 간 연결되어 있는 시스템이다.

② 각 단위 시스템들은 CPU, 메모리 및 운영체제(operating system)를 탑재하여 독립적으로 동작할 수 있도록 한다.

③ 분산 시스템의 전체 상태를 유지하고 관리하는 중앙 서버를 두어야 한다.

④ 활용 분야에 따라 그에 맞는 수준의 투명성(transparency)을 제공할 수 있어야 한다.

정답 3번

출제 포인트

분산 시스템을 구성하는 독립적인 단위 시스템, 네트워크 연결, 투명성의 의미와 중앙 서버의 필수 여부를 묻는 문제이다.

핵심 개념

분산 시스템은 네트워크로 연결된 여러 독립적인 컴퓨터가 통신하고 협력하여 하나의 통합된 시스템처럼 서비스를 제공하는 시스템이다. 각 구성 요소는 독립적으로 동작할 수 있으며, 시스템 구조에 따라 중앙 서버를 사용할 수는 있지만 중앙 서버가 반드시 필요한 것은 아니다. 투명성은 자원이나 처리 위치가 분산되어 있다는 사실을 사용자가 가능한 한 의식하지 않고 시스템을 이용할 수 있게 하는 성질이다.

선지별 해설

① O — 여러 단위 시스템이 네트워크로 연결되어 통신하고 협력한다는 설명은 분산 시스템의 기본 구성과 일치한다.
② O — 각 단위 시스템은 CPU, 메모리, 운영체제를 갖춘 독립적인 컴퓨터로서 자체적으로 동작할 수 있다.
③ X — 틀린 표현은 '전체 상태를 유지하고 관리하는 중앙 서버를 두어야 한다'이다. 중앙 서버를 사용하는 구조도 가능하지만 필수 조건은 아니며, 올바르게는 여러 노드가 역할과 상태를 분산하여 관리할 수 있다고 해야 한다.
④ O — 분산의 복잡성을 사용자에게 감추는 투명성은 활용 목적과 성능·관리 요구에 따라 필요한 수준으로 제공할 수 있다.

오답 포인트

여러 노드를 관리하려면 중앙 서버가 반드시 있어야 한다고 생각하기 쉽다. 그러나 '중앙 서버를 둘 수 있다'와 '중앙 서버를 두어야 한다'는 다르며, 분산 시스템의 본질은 독립적인 노드들이 네트워크를 통해 협력하는 데 있다.

NoteBase

다시 나오면 이렇게 판단

분산 시스템의 특징을 판단할 때는 '독립 노드', '네트워크 통신', '협력', '투명성'을 확인한다. 특정 중앙 장치가 반드시 전체를 통제해야 한다는 표현은 분산 구조의 필수 조건이 아니므로 의심한다.

암기 포인트

분산 시스템은 독립 노드의 네트워크 협력이며, 중앙 서버는 선택 사항이지 필수 사항이 아니다.

NoteBase

문 13. 이진 탐색 트리에 관한 다음 설명 중 가장 옳지 않은 것은?

① 모든 노드는 키를 가지며, 그 키는 유일한 값을 가진다.

② 부모 노드의 왼쪽 서브트리에 있는 모든 노드의 키는 부모노드의 키보다 작다.

③ 자식 노드가 두 개인 경우, 삭제된 노드의 자식을 삭제된 노드의 자리에 위치시킨다.

④ 균형 탐색 트리(balanced search tree)는 최악의 상황에도 탐색 시간 복잡도 $O(\log n)$을 보장한다.

정답 3번

출제 포인트

이 문제는 이진 탐색 트리의 키 배치 규칙과 노드 삭제 방법, 균형 트리의 탐색 시간 복잡도를 묻는 문제이다.

핵심 개념

이진 탐색 트리는 각 노드의 왼쪽 서브트리에는 더 작은 키를, 오른쪽 서브트리에는 더 큰 키를 두어 탐색하는 트리이다. 자식이 둘인 노드를 삭제할 때는 중위 순회 선행자 또는 후속자로 그 노드를 대체한 뒤, 대체에 사용한 노드를 원래 위치에서 삭제한다. 균형 탐색 트리는 높이를 $O(\log n)$으로 유지하여 최악의 경우에도 탐색을 $O(\log n)$에 수행한다.

선지별 해설

① O — 이진 탐색 트리는 각 노드가 키를 가지며, 이 문항이 전제하는 일반적인 정의에서는 중복되지 않는 유일한 키로 노드를 구분한다.
② O — 이진 탐색 트리의 왼쪽 서브트리에 있는 모든 노드의 키는 해당 부모 노드의 키보다 작다. 오른쪽 서브트리의 키는 부모 노드의 키보다 크다.
③ X — 틀린 표현은 '삭제된 노드의 자식을 삭제된 노드의 자리에 위치시킨다'이다. 자식이 둘이면 어느 한 자식을 그대로 올리는 것이 아니라, 왼쪽 서브트리의 최댓값인 중위 선행자 또는 오른쪽 서브트리의 최솟값인 중위 후속자로 삭제 대상 노드를 대체하고 그 대체 노드를 원래 위치에서 삭제한다.
④ O — 균형 탐색 트리는 트리의 높이를 $O(\log n)$ 수준으로 유지하므로 최악의 경우에도 탐색 시간 복잡도 $O(\log n)$을 보장한다.

오답 포인트

③은 자식이 하나인 노드의 삭제 방법과 자식이 둘인 노드의 삭제 방법을 혼동하게 한다. 자식이 하나라면 그 자식을 삭제된 노드의 자리에 연결할 수 있지만, 자식이 둘이면 중위 선행자 또는 중위 후속자를 이용해야 이진 탐색 트리의 순서가 유지된다.

NoteBase

다시 나오면 이렇게 판단

삭제 대상의 자식 수를 먼저 구분한다. 자식이 없으면 바로 삭제하고, 하나이면 그 자식을 연결하며, 둘이면 중위 선행자 또는 중위 후속자로 대체한다.

암기 포인트

BST 삭제는 0개면 삭제, 1개면 자식 연결, 2개면 선행자·후속자 대체이다.

NoteBase

문 14. 기억장치 계층 구조 및 반도체 메모리(DRAM, SRAM)의 특성에 관한 다음 설명 중 옳은 것을 모두 고른 것은?

ㄱ. 보조기억장치(Auxiliary Memory)는 대용량이지만 주기억장치보다 느리고 저렴하다.
ㄴ. SRAM(Static RAM)은 플립플롭(Flip-flop)을 사용하여 재충전(Refresh)이 필요없고 전원 공급 시 데이터가 유지된다.
ㄷ. DRAM(Dynamic RAM)은 SRAM에 비해 속도가 더 빨라서 주로 높은 속도가 필요한 캐시로 사용된다.
ㄹ. 기억장치의 상위 계층(CPU에 가까운 쪽)일수록 접근 속도는 빠르나 용량은 작고 비트 당 가격은 비싸진다.

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

정답 2번

출제 포인트

기억장치 계층에서 속도·용량·가격의 관계와 SRAM·DRAM의 구조 및 용도를 구별하는 문제이다.

핵심 개념

보조기억장치: 주기억장치보다 속도는 느리지만 대용량이고 비트당 가격이 저렴한 비휘발성 저장장치이다.
SRAM: 플립플롭으로 비트를 저장하여 주기적인 재충전이 필요 없고, 빠르지만 집적도가 낮고 비싸므로 주로 캐시에 사용된다. 다만 휘발성 메모리이므로 전원이 공급되는 동안에만 데이터가 유지된다.
DRAM: 커패시터의 전하로 비트를 저장하여 주기적인 재충전이 필요하고, SRAM보다 느리지만 집적도가 높고 저렴하므로 주로 주기억장치에 사용된다.
기억장치 계층: CPU에 가까울수록 접근 속도가 빠르고 용량이 작으며 비트당 가격이 비싸다.

선지별 해설

① X — ㄱ과 ㄴ은 옳지만 ㄹ도 옳으므로 불완전한 조합이다. 틀린 부분은 옳은 설명을 'ㄱ, ㄴ'으로만 고른 것이며, 올바른 조합은 'ㄱ, ㄴ, ㄹ'이다.
② O — ㄱ은 보조기억장치의 대용량·저속·저비용 특성, ㄴ은 SRAM의 플립플롭 방식과 재충전 불필요 특성, ㄹ은 상위 기억장치의 고속·소용량·고비용 특성을 정확히 설명한다.
③ X — ㄷ이 틀리고 옳은 ㄱ이 빠져 있다. DRAM이 SRAM보다 빠른 것이 아니라 SRAM이 DRAM보다 빠르며, 캐시에는 주로 SRAM을 사용한다. 올바른 조합은 'ㄱ, ㄴ, ㄹ'이다.
④ X — ㄷ을 옳은 설명에 포함한 것이 틀렸다. DRAM은 SRAM보다 느리고 주로 주기억장치에 사용되며, 올바른 조합은 'ㄱ, ㄴ, ㄹ'이다.

NoteBase

오답 포인트

ㄷ의 ‘DRAM이 SRAM보다 빠르다’와 ‘DRAM이 캐시에 사용된다’는 표현이 서로 뒤바뀐 함정이다. 빠르고 재충전이 필요 없는 SRAM이 캐시에 사용되고, 상대적으로 느리지만 고집적·저비용인 DRAM이 주기억장치에 사용된다.

다시 나오면 이렇게 판단

먼저 SRAM은 플립플롭·재충전 불필요·고속·캐시, DRAM은 커패시터·재충전 필요·상대적 저속·주기억장치로 구분한다. 이어서 기억장치가 CPU에 가까워질수록 속도와 비트당 가격은 증가하고 용량은 감소하는지 확인한다.

암기 포인트

SRAM은 빠른 캐시, DRAM은 큰 주기억장치이다. 위로 갈수록 빠르고 작고 비싸다.

NoteBase

문 15. 최신 ICT 기술 및 패러다임에 관한 다음 설명 중 가장 옳지 않은 것은?

① 공간 컴퓨팅(Spatial Computing)은 가상·증강현실 기술을 기반으로 디지털 정보와 물리적 공간을 결합하여 상호작용하는 기술이다.

② 대규모 언어모델(Large Language Model)은 방대한 텍스트를 학습하여 인간 수준의 자연스러운 언어 생성과 추론을 수행한다.

③ 소버린 클라우드(Sovereign Cloud)는 국가적 데이터 주권을 보호하기 위해 해당 국가의 법령과 규제 요건을 엄격히 준수하는 모델이다.

④ 인물 합성 기술(Deepfake)은 생성형 대립 신경망(GAN)을 활용하며, 두 신경망이 협력하여 데이터의 평균값으로 인물을 합성한다.

정답 4번

출제 포인트

최신 ICT 개념의 핵심 정의를 구별하고, GAN의 생성자와 판별자가 작동하는 관계를 판단하는 문제이다.

핵심 개념

공간 컴퓨팅: 디지털 정보와 현실의 물리적 공간을 결합하여 사용자가 공간 속에서 상호작용하게 하는 기술이다.
대규모 언어모델: 방대한 텍스트에서 언어의 패턴과 관계를 학습하여 자연어를 이해·생성하고 추론 과제를 수행하는 모델이다.
소버린 클라우드: 데이터 주권을 확보하기 위해 특정 국가의 법률·규제와 데이터 통제 요건을 따르도록 설계된 클라우드 모델이다.
딥페이크: 인공지능으로 사람의 얼굴·음성 등을 실제처럼 합성하거나 변형하는 기술이며, GAN을 이용할 때 생성자와 판별자의 적대적 학습으로 결과를 개선한다.

선지별 해설

① O — 공간 컴퓨팅은 가상현실·증강현실 등의 기술을 활용하여 디지털 정보와 물리적 공간을 결합하고 공간 기반 상호작용을 구현한다.
② O — 대규모 언어모델은 방대한 텍스트를 학습하여 자연스러운 언어 생성과 문맥에 따른 추론을 수행한다.
③ O — 소버린 클라우드는 국가의 데이터 주권을 보호하기 위해 해당 국가의 법령과 규제 요건을 준수하고 데이터의 저장·처리·통제 범위를 관리하는 모델이다.
④ X — 틀린 표현은 '두 신경망이 협력하여 데이터의 평균값으로 인물을 합성한다'이다. GAN에서는 생성자가 가짜 데이터를 만들고 판별자가 진짜와 가짜를 구별하며, 두 신경망이 서로 경쟁하는 적대적 학습을 통해 합성 결과를 정교하게 만든다.

NoteBase

오답 포인트

④는 GAN이 두 신경망으로 구성된다는 점은 맞지만, 그 관계를 '협력'과 '평균값 합성'으로 바꾸었다. 생성자와 판별자의 핵심 관계는 경쟁이며, 단순히 데이터의 평균값으로 인물을 만드는 방식이 아니다.

다시 나오면 이렇게 판단

ICT 용어 문항에서는 각 기술의 목적과 작동 원리를 함께 확인한다. GAN은 '생성자는 만들고, 판별자는 가려내며, 둘은 적대적으로 학습한다'는 구조를 기준으로 판단한다.

암기 포인트

GAN은 생성자 대 판별자, 협력이 아니라 적대적 경쟁이다.

NoteBase

문 16. 한 시스템에 3개의 프로세스 P1, P2, P3가 존재하며, 이들이 준비 큐(ready queue)에 도착하는 시간이 각각 0, 1, 4이고, 이 때 이들이 사용할 CPU burst time이 각각 5, 4, 2라 가정한다. 시간 할당량(time quantum) 3인 RR(Round-Robin) 스케줄링 기법을 사용할 경우 이들이 CPU burst를 모두 완료할 때까지 CPU를 할당받는 순서로 옳은 것은?

① P1 → P2 → P3

② P3 → P2 → P1

③ P1 → P2 → P3 → P1 → P2

④ P1 → P2 → P1 → P3 → P2

정답 4번

출제 포인트

시간 할당량이 3인 라운드 로빈에서 프로세스의 도착 시각과 남은 CPU 버스트를 함께 반영하여 준비 큐의 실행 순서를 구하는 문제이다.

올바른 순서

1. P1: 0~3에 실행되어 버스트 5 중 3을 사용하고 2가 남는다.
2. P2: P1 실행 중 시각 1에 도착하므로 3~6에 실행되어 버스트 4 중 3을 사용하고 1이 남는다. P3는 이 실행 중 시각 4에 도착한다.
3. P1: 시각 3에 준비 큐의 뒤로 이동했으므로 P3보다 앞에 있고, 6~8에 남은 2를 실행하여 종료한다.
4. P3: 8~10에 버스트 2를 모두 실행하여 종료한다.
5. P2: 10~11에 남은 1을 실행하여 종료한다.

따라서 CPU 할당 순서는 P1 → P2 → P1 → P3 → P2이다. P3가 시각 4에 도착했을 때 P1은 이미 시각 3부터 준비 큐에서 기다리고 있으므로 P1이 P3보다 먼저 실행된다.

선지별 해설

① X — P1 → P2 → P3만으로는 P1의 남은 버스트 2와 P2의 남은 버스트 1에 대한 재할당이 빠져 있다. 올바르게는 P1 → P2 → P1 → P3 → P2이다.

② X — 시각 0에는 P1만 도착해 있으므로 P3부터 실행할 수 없다. 올바르게는 P1이 먼저 실행되며 전체 순서는 P1 → P2 → P1 → P3 → P2이다.

③ X — P2의 첫 시간 할당이 끝나는 시각 6의 준비 큐에서는 시각 3부터 기다린 P1이 시각 4에 도착한 P3보다 앞선다. 따라서 P3보다 P1이 먼저 실행되어야 하며, 올바르게는 P1 → P2 → P1 → P3 → P2이다.

④ O — 각 시간 할당이 끝날 때 미완료 프로세스를 준비 큐 뒤로 보내고 새 도착 프로세스를 도착 시각에 삽입하면 P1 → P2 → P1 → P3 → P2가 된다.

NoteBase

오답 포인트

③과 ④의 차이는 P1과 P3의 선후관계이다. P1은 시각 3에 첫 실행을 마치고 준비 큐로 돌아가며, P3는 그보다 늦은 시각 4에 도착하므로 P1이 앞선다. 또한 시간 할당량보다 버스트가 큰 P1과 P2는 한 번의 CPU 할당으로 끝나지 않는다.

다시 나오면 이렇게 판단

현재 시각마다 준비 큐의 맨 앞 프로세스를 최대 시간 할당량만큼 실행하고, 실행 중 새로 도착한 프로세스는 큐 뒤에 넣는다. 실행이 끝나지 않은 프로세스도 시간 할당 종료 시 큐 뒤로 보내며, 매 구간 뒤의 준비 큐 순서를 기록하면 된다.

암기 포인트

RR은 선입선출 준비 큐를 순환한다. 먼저 기다린 프로세스가 먼저이며, 미완료 프로세스는 시간 할당 후 맨 뒤로 이동한다.

NoteBase

문 17. 인공지능 모델의 데이터 처리 방식에 관한 다음 설명 중 가장 옳지 않은 것은?

① 합성곱신경망(CNN)은 이미지의 공간적 특징(spatial feature)을 추출하는 데에 최적화되어 있어 객체 인식 등에 주로 사용된다.

② 트랜스포머(transformer)는 문장 내 단어 간의 관계를 파악하는 어텐션(attention) 메커니즘을 통해 자연어처리 분야에서 널리 사용된다.

③ 순환신경망(RNN)은 병렬처리에 효율적이어서 대규모 시계열 데이터를 빠르게 처리하는 데에 널리 사용된다.

④ 멀티모달(multimodal) AI는 텍스트, 이미지, 오디오 등 서로 다른 형태의 데이터를 통합적으로 학습하여 복합적인 정보를 처리한다.

정답 3번

출제 포인트

CNN, 트랜스포머, RNN, 멀티모달 AI가 각각 어떤 데이터와 처리 구조에 적합한지를 구별하는 문제이다.

핵심 개념

CNN은 합성곱 연산으로 이미지의 국소적·공간적 특징을 추출하는 신경망이다.
트랜스포머는 어텐션으로 입력 요소 사이의 관계를 파악하며, 순차 계산 의존성이 낮아 병렬 처리에 유리한 모델이다.
RNN은 이전 시점의 은닉 상태를 다음 시점 계산에 사용하는 순환 구조로 순서가 있는 데이터를 처리하지만, 시점 간 의존성 때문에 병렬화가 어렵다.
멀티모달 AI는 텍스트·이미지·오디오처럼 서로 다른 양식의 데이터를 함께 학습하고 처리한다.

선지별 해설

① O — CNN은 필터를 이용해 이미지의 국소 패턴과 공간적 특징을 추출하므로 객체 인식 등에 널리 사용된다.
② O — 트랜스포머의 어텐션은 문장 안의 단어들이 서로 어떤 관계를 갖는지 반영하므로 자연어 처리에 널리 활용된다.
③ X — 틀린 표현은 '병렬처리에 효율적'이라는 부분이다. RNN은 앞 시점의 계산 결과가 다음 시점 계산에 이어지는 순차 구조이므로 시점 단위 병렬화가 어렵다. 올바르게는 'RNN은 순차 데이터를 처리하는 데 적합하지만 시점 간 의존성 때문에 병렬 처리에는 비효율적이다'라고 해야 한다.
④ O — 멀티모달 AI는 텍스트, 이미지, 오디오 등 여러 데이터 양식을 통합하여 복합적인 정보를 학습하고 처리한다.

NoteBase

오답 포인트

③은 RNN이 시계열 데이터에 사용된다는 사실에 '병렬처리에 효율적'이라는 반대 특성을 섞었다. 순서가 있는 데이터를 다룰 수 있다는 것과 여러 시점을 동시에 효율적으로 계산할 수 있다는 것은 다른 의미이다.

다시 나오면 이렇게 판단

모델과 처리 방식을 연결해 판단한다. CNN은 공간 특징, 트랜스포머는 어텐션과 병렬 처리, RNN은 순차 의존성, 멀티모달 AI는 서로 다른 데이터 양식의 통합이 핵심이다.

암기 포인트

CNN은 공간, 트랜스포머는 어텐션·병렬, RNN은 순차·병렬화 제약, 멀티모달은 여러 양식의 통합이다.

NoteBase

문 18. 소프트웨어 생명 주기(SDLC) 모형 중 나선형(Spiral) 모형의 특징으로 다음 설명 중 가장 옳은 것은?

① 시스템의 핵심 기능을 실험적으로 만들어 사용자에게 보여주고 평가하게 한다.

② 개발의 각 단계 사이에 결과물이 있어 명확히 구분되며 이전 작업으로 돌아가는 재작업이 없다.

③ 제약조건 결정, 위험 요소 분석 및 해결, 개발과 평가, 다음 단계 계획의 4단계를 반복하며 개발한다.

④ 프로젝트에 대한 피드백을 빨리 받고 방향을 평가하기 위해 짧은 사이클을 반복하며 실행 가능한 소프트웨어를 개발한다.

정답 3번

출제 포인트

나선형 모형의 핵심인 위험 분석과 반복 개발을 다른 소프트웨어 생명 주기 모형의 특징과 구별하는 문제이다.

핵심 개념

나선형 모형은 개발 주기를 반복하면서 매 주기마다 목표와 제약조건을 정하고, 위험을 분석·해결한 뒤 개발과 평가를 수행하고 다음 반복을 계획하는 위험 중심 모형이다.
프로토타이핑 모형은 시제품을 만들어 사용자의 요구를 확인하고, 폭포수 모형은 단계를 순차적으로 진행하며, 애자일 방식은 짧은 반복 주기로 실행 가능한 소프트웨어를 제공하고 피드백을 반영한다.

선지별 해설

① X — 핵심 기능을 실험적으로 구현한 시제품을 사용자에게 보여 주고 평가받는 것은 프로토타이핑 모형의 특징이다. 틀린 표현은 이를 나선형 모형의 특징으로 본 부분이며, 올바르게는 '시제품을 통해 사용자 요구를 확인하는 프로토타이핑 모형의 특징'이다.
② X — 단계별 결과물이 명확하고 순차적으로 진행되는 것은 폭포수 모형의 특징이다. 특히 '이전 작업으로 돌아가는 재작업이 없다'는 표현이 부정확하며, 올바르게는 '폭포수 모형은 단계 간 되돌아가기가 어렵고 변경 비용이 크다'이다.
③ O — 제약조건 결정, 위험 요소 분석 및 해결, 개발과 평가, 다음 단계 계획을 반복하는 것은 나선형 모형의 대표적인 진행 방식이다. 위험을 반복 주기마다 분석하고 줄인다는 점이 핵심이다.
④ X — 짧은 사이클마다 실행 가능한 소프트웨어를 개발하여 빠르게 피드백을 받는 것은 애자일 방식의 특징이다. 틀린 표현은 이를 나선형 모형의 고유 특징으로 본 부분이며, 올바르게는 '짧은 반복 주기로 결과물을 제공하고 피드백을 반영하는 애자일 방식의 특징'이다.

오답 포인트

나선형 모형도 반복 개발을 하므로 ④의 '짧은 사이클 반복'에 끌리기 쉽다. 그러나 나선형 모형을 식별하는 결정적 단서는 반복 자체가 아니라 각 주기에서 수행하는 '위험 요소 분석 및 해결'이다.

NoteBase

다시 나오면 이렇게 판단

선지에서 '위험 분석'과 '다음 반복 계획'이 함께 제시되면 나선형 모형으로 판단한다. '시제품과 사용자 평가'는 프로토타이핑, '단계별 순차 진행'은 폭포수, '짧은 주기와 실행 가능한 소프트웨어 및 빠른 피드백'은 애자일 방식으로 구별한다.

암기 포인트

나선형은 위험 중심 반복, 프로토타이핑은 시제품, 폭포수는 순차 단계, 애자일은 짧은 주기와 빠른 피드백이다.

NoteBase

문 19. 소프트웨어 유지보수의 4가지 유형 중 다음 <보기>에 해당하는 작업으로 가장 옳은 것은?

<보기>
한 기업에서 사용 중인 급여 관리 시스템이 기존 Windows 10 환경에서 최신 운영체제인 Windows 11로 업그레이드됨에 따라, 시스템이 새로운 운영체제 환경에서도 오류 없이 정상적으로 동작할 수 있도록 내부 코드를 수정하고 환경 설정을 변경하였다.

① 수정(Corrective) 유지보수
② 적응(Adaptive) 유지보수
③ 완전(Perfective) 유지보수
④ 예방(Preventive) 유지보수

정답 2번

출제 포인트

소프트웨어를 변경한 원인이 결함, 환경 변화, 성능·기능 개선, 미래의 장애 예방 중 무엇인지에 따라 유지보수 유형을 구분하는 문제이다.

핵심 개념

수정 유지보수는 운영 중 발견된 오류나 결함을 고치는 작업이다.
적응 유지보수는 운영체제, 하드웨어, 법규 등 외부 환경의 변화에 맞추어 소프트웨어를 변경하는 작업이다.
완전 유지보수는 사용자의 요구에 따라 기능, 성능, 사용 편의성 등을 개선하는 작업이다.
예방 유지보수는 앞으로 발생할 수 있는 장애를 줄이도록 구조와 유지보수성을 미리 개선하는 작업이다.

선지별 해설

① X — '수정(Corrective) 유지보수'가 틀린 표현이다. 수정 유지보수는 이미 발견된 오류나 결함을 바로잡는 작업이며, 이 사례처럼 운영체제 변화에 대응하는 작업은 적응 유지보수이다.
② O — Windows 10에서 Windows 11로 운영체제 환경이 바뀌었고, 새 환경에서도 정상 동작하도록 코드와 설정을 변경했으므로 적응(Adaptive) 유지보수이다.
③ X — '완전(Perfective) 유지보수'가 틀린 표현이다. 완전 유지보수는 기능 추가, 성능 향상, 사용 편의 개선처럼 소프트웨어의 완성도를 높이는 작업이며, 운영체제 변화에 맞춘 작업은 적응 유지보수이다.
④ X — '예방(Preventive) 유지보수'가 틀린 표현이다. 예방 유지보수는 잠재적 결함이나 향후 장애를 줄이기 위해 코드 구조와 유지보수성을 미리 개선하는 작업이며, 외부 실행 환경의 변화에 대응하는 작업은 적응 유지보수이다.

NoteBase

오답 포인트

‘오류 없이 정상적으로 동작’이라는 결과만 보고 수정 또는 예방 유지보수로 판단하기 쉽다. 그러나 분류 기준은 작업 결과가 아니라 변경을 일으킨 원인이며, 이 사례의 직접적인 원인은 Windows 11이라는 외부 환경의 변화이다.

다시 나오면 이렇게 판단

변경 원인을 먼저 확인한다. 이미 생긴 결함은 수정, 운영 환경의 변화는 적응, 기능·성능 개선은 완전, 미래 장애의 사전 방지는 예방 유지보수로 판단한다.

암기 포인트

결함은 수정, 환경은 적응, 개선은 완전, 사전 대비는 예방이다.

NoteBase

문 20. 머신러닝(Machine Learning)의 학습 방식에 관한 다음 설명 중 가장 옳은 것은?

① 지도학습은 정답(Label)이 없는 데이터를 분석하여 데이터 내부의 유용한 패턴이나 군집을 찾는다.

② 비지도학습의 대표적인 응용은 데이터를 여러 범주 중 하나로 나누는 분류와 연속적인 값을 예측하는 회귀 분석이다.

③ 전이학습은 라벨링이 된 대량의 데이터로만 처음부터 끝까지 학습시키는 방법이다.

④ 강화학습은 환경과 상호작용하여 보상을 최대화하는 방향으로 스스로 학습하는 방식이다.

정답 4번

출제 포인트

머신러닝의 지도학습·비지도학습·전이학습·강화학습을 학습 데이터와 학습 방식에 따라 구별하는 문제이다.

핵심 개념

지도학습: 입력과 정답(Label)이 함께 있는 데이터로 입력과 정답의 관계를 학습하며, 대표 과제는 분류와 회귀이다.
비지도학습: 정답이 없는 데이터에서 구조, 패턴, 군집 등을 찾는다.
전이학습: 한 작업에서 미리 학습한 모델의 지식을 관련된 다른 작업에 재사용하여 학습한다.
강화학습: 에이전트가 환경과 상호작용하면서 받은 보상을 바탕으로 누적 보상을 크게 하는 행동 전략을 학습한다.

선지별 해설

① X — '지도학습은 정답(Label)이 없는 데이터를 분석한다'는 부분이 틀렸다. 정답이 없는 데이터에서 패턴이나 군집을 찾는 것은 비지도학습이며, 지도학습은 정답이 붙은 데이터로 학습한다.
② X — '비지도학습의 대표적인 응용은 분류와 회귀'라는 부분이 틀렸다. 분류와 회귀는 정답을 이용하는 지도학습의 대표 과제이고, 비지도학습의 대표 과제에는 군집화 등이 있다.
③ X — '라벨링이 된 대량의 데이터로만 처음부터 끝까지 학습'한다는 부분이 틀렸다. 전이학습은 이미 학습된 모델의 지식을 새로운 관련 작업에 옮겨 활용하는 방법이므로 처음부터 전 과정을 다시 학습하는 방식과 구별된다.
④ O — 강화학습은 환경과 상호작용하며 행동의 결과로 보상을 받고, 그 보상을 최대화하는 방향으로 행동 전략을 학습한다.

오답 포인트

①과 ②는 지도학습과 비지도학습의 특징을 서로 바꾸어 놓았고, ③은 전이학습의 핵심인 '기존 학습 지식의 재사용'을 '처음부터 학습'으로 뒤집었다.

NoteBase

다시 나오면 이렇게 판단

정답(Label)이 있으면 지도학습, 정답 없이 패턴·군집을 찾으면 비지도학습, 기존 모델의 지식을 재사용하면 전이학습, 환경과 상호작용하며 보상을 최대화하면 강화학습으로 판단한다.

암기 포인트

지도는 정답, 비지도는 패턴, 전이는 재사용, 강화는 보상이다.

NoteBase

문 21. 다음 릴레이션 학생(R)과 수강(S)이 주어졌을 때, 아래 관계대수식을 수행한 최종 결과값으로 옳은 것은?

[학생(R) 릴레이션]

학번	이름	학과
101	김영희	컴퓨터
102	박철수	전자
103	홍길동	컴퓨터

[수강(S) 릴레이션]

학번	과목	성적
101	DB	A
101	OS	B
102	DB	A
103	DB	B

_이름≤ft(_학과 = '컴퓨터' AND 과목 = 'DB'≤ft(R _R.학번 = S.학번 S))

① {김영희}

② {김영희, 박철수}

③ {김영희, 홍길동}

④ {김영희, 박철수, 홍길동}

정답 3번

출제 포인트

학생 릴레이션과 수강 릴레이션을 학번으로 조인한 뒤, 학과와 과목 조건을 모두 만족하는 튜플만 선택하고 이름 속성만 추출하는 관계대수 연산을 해석하는 문제이다.

자료 읽는 순서

1. 학생 *R*과 수강 *S*를 R.학번=S.학번 조건으로 조인한다.
2. 조인 결과에서 학과가 '컴퓨터'이면서 과목이 'DB'인 튜플만 선택한다. AND 조건이므로 두 조건을 동시에 만족해야 한다.
3. 남은 튜플에서 이름 속성만 프로젝션한다.

NoteBase

핵심 근거

학번 101 김영희는 컴퓨터학과이고 DB를 수강하므로 남는다. 학번 102 박철수는 DB를 수강하지만 전자학과이므로 제외된다. 학번 103 홍길동은 컴퓨터학과이고 DB를 수강하므로 남는다. 따라서 이름을 프로젝션한 최종 결과는 {김영희, 홍길동}이다.

선지별 해설

① X — {김영희}는 학생 릴레이션의 학번 103 행과 수강 릴레이션의 학번 103·DB 행에서 확인되는 홍길동을 누락했다. 올바르게는 {김영희, 홍길동}이다.
② X — {김영희, 박철수}에서 박철수는 학생 릴레이션상 전자학과이므로 '학과=컴퓨터' 조건을 만족하지 않으며, 조건을 만족하는 홍길동도 빠졌다. 올바르게는 {김영희, 홍길동}이다.
③ O — 김영희와 홍길동은 각각 학번 101과 103이며, 두 사람 모두 학생 릴레이션에서 컴퓨터학과이고 수강 릴레이션에서 DB 수강이 확인된다. 따라서 {김영희, 홍길동}이 최종 결과이다.
④ X — {김영희, 박철수, 홍길동}에는 전자학과인 박철수가 잘못 포함되었다. 올바르게는 박철수를 제외한 {김영희, 홍길동}이다.

오답 포인트

DB 수강자만 찾으면 박철수까지 포함되지만, 선택 조건은 '학과=컴퓨터 AND 과목=DB'이다. 과목 조건만 확인하거나 AND를 OR처럼 처리하는 것이 함정이다.

다시 나오면 이렇게 판단

관계대수식은 안쪽 연산부터 처리한다. 먼저 조인 조건으로 릴레이션을 결합하고, 선택 σ로 행을 거른 뒤, 프로젝션 π로 요구된 열만 남긴다.

암기 포인트

조인은 표 결합, 선택 σ는 행 추출, 프로젝션 π는 열 추출이다.

NoteBase

문 22. 다음 정렬(sorting) 기법들 중 평균 시간 복잡도(average time complexity)가 $O(n \cdot \log_2 n)$이 아닌 기법은?

① 병합 정렬(merge sort)

② 힙 정렬(heap sort)

③ 삽입 정렬(insertion sort)

④ 퀵 정렬(quick sort)

정답 3번

출제 포인트

대표적인 비교 정렬의 평균 시간 복잡도를 구분하는 문제이다.

핵심 개념

시간 복잡도는 입력 크기 n이 커질 때 알고리즘의 실행 시간이 증가하는 정도를 나타낸다.
병합 정렬, 힙 정렬, 퀵 정렬의 평균 시간 복잡도는 $O(n\log_2 n)$이다. 삽입 정렬의 평균 시간 복잡도는 $O(n^2)$이다.

선지별 해설

① X — 병합 정렬은 배열을 계속 분할한 뒤 정렬된 부분들을 병합하므로 평균 시간 복잡도가 $O(n\log_2 n)$이다. 따라서 '평균 시간 복잡도가 $O(n\log_2 n)$이 아닌 기법'으로 고른 것이 틀리며, 올바르게는 평균 시간 복잡도가 $O(n\log_2 n)$인 기법이다.
② X — 힙 정렬은 힙을 구성한 뒤 최댓값 또는 최솟값을 반복하여 꺼내므로 평균 시간 복잡도가 $O(n\log_2 n)$이다. 따라서 이 선택은 틀리며, 올바르게는 평균 시간 복잡도가 $O(n\log_2 n)$인 기법이다.
③ O — 삽입 정렬은 각 원소를 앞의 정렬된 부분에 삽입하는 과정에서 평균적으로 원소들을 여러 번 비교하고 이동하므로 평균 시간 복잡도가 $O(n^2)$이다. 따라서 조건에 해당한다.
④ X — 퀵 정렬은 평균적으로 피벗이 입력을 균형 있게 나누어 평균 시간 복잡도가 $O(n\log_2 n)$이다. 따라서 이 선택은 틀리며, 올바르게는 평균 시간 복잡도가 $O(n\log_2 n)$인 기법이다.

오답 포인트

퀵 정렬은 피벗 분할이 계속 한쪽으로 치우치면 최악 시간 복잡도가 $O(n^2)$이지만, 문제는 최악이 아니라 평균 시간 복잡도를 묻는다. 이 최악의 경우를 떠올려 ④를 고르는 것이 핵심 함정이다.

다시 나오면 이렇게 판단

먼저 평균·최선·최악 중 어느 경우를 묻는지 확인한다. 평균 기준에서는 병합·힙·퀵 정렬을 $O(n\log n)$, 삽입 정렬을 $O(n^2)$으로 구분한다.

NoteBase

암기 포인트

평균 기준으로 병합·힙·퀵은 $n\log n$, 삽입은 n^2이다.

문 23. Python 언어의 리스트(list), 튜플(tuple), 집합(set) 자료형에 관한 다음 설명 중 가장 옳지 않은 것은?

① [1, 2, 3]은 리스트에 해당하며 (1, 2, 3)은 튜플에 해당한다.

② 하나의 리스트가 다른 리스트의 원소가 될 수 있다.

③ 리스트나 튜플에서는 인덱싱(indexing)이 가능하지만 집합에서는 인덱싱이 불가능하다.

④ 리스트의 원소값은 변경할 수 없으나 튜플의 원소값은 변경할 수 있다.

정답 4번

출제 포인트

Python의 리스트·튜플·집합을 표기법, 중첩 가능 여부, 인덱싱, 원소 변경 가능 여부로 구분하는 문제이다.

핵심 개념

리스트: 순서가 있는 가변 자료형으로, 대괄호로 표현하며 인덱싱과 원소 변경이 가능하다.
튜플: 순서가 있는 불변 자료형으로, 일반적으로 소괄호와 쉼표로 표현하며 인덱싱은 가능하지만 원소 변경은 불가능하다.
집합: 중복을 허용하지 않는 비순서형 자료형으로, 특정 위치를 전제로 하는 인덱싱은 불가능하다.

선지별 해설

① O — [1, 2, 3]은 대괄호를 사용한 리스트이고, (1, 2, 3)은 쉼표로 원소를 구분한 튜플이다.
② O — 리스트는 다른 객체를 원소로 담을 수 있으므로 하나의 리스트를 다른 리스트의 원소로 넣을 수 있다. 예를 들어 [1, [2, 3]]은 중첩 리스트이다.
③ O — 리스트와 튜플은 원소의 순서가 있어 위치 번호로 인덱싱할 수 있다. 집합은 원소의 위치나 순서를 보장하지 않으므로 인덱싱할 수 없다.
④ X — 틀린 표현은 '리스트의 원소값은 변경할 수 없으나 튜플의 원소값은 변경할 수 있다'이다. 실제로는 리스트의 원소값은 변경할 수 있고, 튜플의 원소값은 생성 후 변경할 수 없다.

오답 포인트

④는 리스트와 튜플의 가변성을 정확히 뒤바꾼 선지이다. 둘 다 인덱싱은 가능하지만, 인덱스로 접근한 원소를 새 값으로 바꿀 수 있는 것은 리스트뿐이다.

다시 나오면 이렇게 판단

먼저 순서의 유무로 인덱싱 가능성을 판단하고, 다음으로 가변 여부를 구분한다. 리스트는 '순서 있음·변경 가능', 튜플은 '순서 있음·변경 불가', 집합은 '순서 없음·인덱싱 불가'이다.

NoteBase

암기 포인트

리스트는 바꿀 수 있고, 튜플은 고정되며, 집합에는 위치 번호가 없다.

NoteBase

문 24. 다음 데이터베이스 트랜잭션의 4가지 특성(ACID) 중 “트랜잭션의 실행이 데이터베이스에 모두 반영되든지 아니면 전혀 반영되지 않아야 한다.”는 성질로 옳은 것은?

① 원자성(Atomicity)

② 일관성(Consistency)

③ 고립성(Isolation)

④ 지속성(Durability)

정답 1번

출제 포인트

트랜잭션의 ACID 특성 중 작업 전체를 하나의 단위로 처리하여 모두 반영하거나 모두 취소하는 성질을 구별하는 문제이다.

핵심 개념

원자성(Atomicity): 트랜잭션의 연산을 더 나눌 수 없는 하나의 작업 단위로 취급하여 전부 실행하거나 전부 취소하는 성질이다. 키워드는 'all or nothing'이다.
일관성(Consistency): 트랜잭션 수행 전후에 데이터베이스의 무결성 제약과 규칙이 유지되어 유효한 상태에서 다른 유효한 상태로 바뀌는 성질이다.
고립성(Isolation): 동시에 실행되는 트랜잭션들이 서로의 중간 처리 결과에 간섭하지 않도록 하는 성질이다.
지속성(Durability): 성공적으로 완료된 트랜잭션의 결과가 장애가 발생하더라도 영구적으로 보존되는 성질이다.

선지별 해설

① O — 원자성은 트랜잭션의 모든 연산이 전부 반영되거나 하나도 반영되지 않아야 한다는 성질이므로 제시된 설명과 일치한다.
② X — 틀린 부분은 제시된 설명을 일관성과 연결한 것이다. 일관성은 데이터베이스의 무결성 제약과 규칙을 유지하는 성질이며, 제시된 설명에 해당하는 올바른 개념은 원자성이다.
③ X — 틀린 부분은 제시된 설명을 고립성과 연결한 것이다. 고립성은 동시에 실행되는 트랜잭션들이 서로의 중간 결과에 간섭하지 않게 하는 성질이며, 제시된 설명에 해당하는 올바른 개념은 원자성이다.
④ X — 틀린 부분은 제시된 설명을 지속성과 연결한 것이다. 지속성은 완료된 트랜잭션의 결과가 장애 후에도 보존되는 성질이며, 제시된 설명에 해당하는 올바른 개념은 원자성이다.

NoteBase

오답 포인트

'데이터베이스에 반영된다'는 표현만 보고 완료 결과의 보존을 뜻하는 지속성을 고르기 쉽다. 그러나 이 문항의 결정적 표현은 '모두 반영되든지 아니면 전혀 반영되지 않아야 한다'이며, 이는 전부 또는 전무를 뜻하는 원자성이다.

다시 나오면 이렇게 판단

전부 수행 또는 전부 취소이면 원자성, 무결성 규칙 유지이면 일관성, 동시 실행 간 간섭 방지이면 고립성, 완료 결과의 영구 보존이면 지속성으로 판단한다.

암기 포인트

A는 All or nothing, C는 제약 유지, I는 서로 격리, D는 완료 결과 보존이다.

NoteBase

문 25. 암 진단 모델에서 실제 암 환자 100명 중 90명을 암 환자로 찾아냈고, 정상인 100명 중 20명을 암 환자로 오진했다면, 이 모델의 재현율(recall)과 정밀도(precision)로 옳은 것은? (단, 결과 수치는 소수점 셋째 자리에서 반올림하여 표기함)

① 재현율: 0.90, 정밀도: 0.82

② 재현율: 0.90, 정밀도: 0.85

③ 재현율: 0.85, 정밀도: 0.90

④ 재현율: 0.82, 정밀도: 0.90

정답 1번

출제 포인트

혼동행렬의 참양성(TP), 거짓음성(FN), 거짓양성(FP)을 구분하여 재현율과 정밀도를 계산하는 문제이다.

조건 정리

실제 암 환자 100명 중 90명을 암 환자로 판정했으므로 TP=90, FN=10이다. 정상인 100명 중 20명을 암 환자로 오진했으므로 FP=20이다. 재현율은 실제 양성 중 찾아낸 비율이고, 정밀도는 양성으로 판정한 사람 중 실제 양성의 비율이다.

사용할 공식

재현율은 Recall $= \frac{TP}{TP+FN}$이고, 정밀도는 Precision $= \frac{TP}{TP+FP}$이다. 재현율의 분모는 실제 암 환자 전체이고, 정밀도의 분모는 모델이 암 환자라고 판정한 전체이므로 이 공식을 사용한다.

계산 과정

1. 재현율은 $\frac{90}{90+10} = 0.90$이다.
2. 정밀도는 $\frac{90}{90+20} = \frac{90}{110} \approx 0.81818$이다.
3. 소수점 셋째 자리에서 반올림하면 정밀도는 0.82이다. 따라서 재현율은 0.90, 정밀도는 0.82이다.

선지 판단

① O — 재현율 0.90과 정밀도 0.82가 모두 계산 결과와 일치한다.
② X — ‘정밀도 0.85’가 틀렸다. 올바르게는 정밀도 0.82이다.
③ X — ‘재현율 0.85, 정밀도 0.90’이 틀렸다. 올바르게는 재현율 0.90, 정밀도 0.82이다.
④ X — ‘재현율 0.82, 정밀도 0.90’이 틀렸다. 올바르게는 재현율 0.90, 정밀도 0.82이다.

NoteBase

오답 포인트

정상인을 암 환자로 오진한 20명은 FP이므로 재현율의 분모가 아니라 정밀도의 분모에 더한다. 재현율과 정밀도의 분모를 바꾸면 ③·④처럼 두 값을 뒤섞게 된다. 또한 90/110은 약 0.81818이므로 소수점 셋째 자리에서 반올림한 값은 0.82이며 0.85가 아니다.

다시 나오면 이렇게 판단

먼저 TP·FN·FP를 확정한 뒤, 재현율은 '실제 양성 전체'를 분모로 하고 정밀도는 '양성 판정 전체'를 분모로 하여 계산한다. 마지막에 문제에서 지정한 자리에서 반올림한다.

암기 포인트

재현율은 실제 양성을 얼마나 놓치지 않았는지, 정밀도는 양성 판정이 얼마나 정확했는지를 나타낸다.

NoteBase