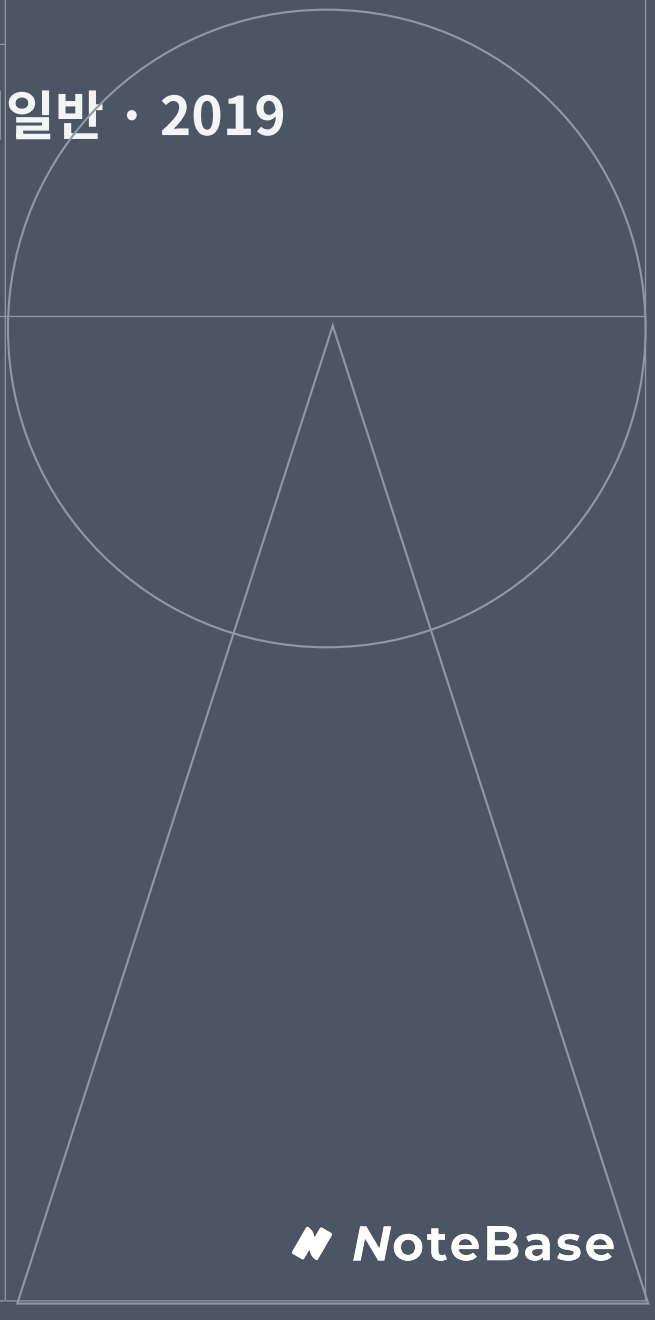
기출OX 해설집

공무원/국회직/9급 · 컴퓨터일반 · 2019



NoteBase

기출OX 해설집

공무원/국회직/9급 · 컴퓨터일반 · 2019

공무원/국회직/9급 · 컴퓨터일반 · 2019

1. 다음 <보기>의 저장장치 중 접근(access) 속도가 빠른 것부터 순서대로 나열한 것으로 옳은 것은?

<보 기>

ㄱ. 레지스터	ㄴ. 주기억장치
ㄷ. 캐시메모리	ㄹ. 하드디스크

① ㄱ, ㄴ, ㄷ, ㄹ
② ㄱ, ㄷ, ㄴ, ㄹ
③ ㄱ, ㄷ, ㄹ, ㄴ
④ ㄷ, ㄱ, ㄴ, ㄹ
⑤ ㄷ, ㄱ, ㄹ, ㄴ

정답 2번

출제 포인트

저장장치 계층에서 CPU와의 거리와 접근 속도를 기준으로 레지스터, 캐시메모리, 주기억장치, 하드디스크의 순서를 구분하는 문제이다.

핵심 개념

레지스터: CPU 내부에서 연산에 필요한 데이터나 명령을 잠시 보관하는 가장 빠른 저장공간이다.
캐시메모리: CPU와 주기억장치 사이의 속도 차이를 줄이기 위해 자주 쓰는 데이터를 임시 저장하는 고속 메모리이다.
주기억장치: 실행 중인 프로그램과 데이터를 CPU가 사용할 수 있도록 저장하는 메모리이다.
하드디스크: 데이터를 장기간 보관하는 보조기억장치로, 이 보기의 장치 중 접근 속도가 가장 느리다.
접근 속도가 빠른 순서는 레지스터(ㄱ) → 캐시메모리(ㄷ) → 주기억장치(ㄴ) → 하드디스크(ㄹ)이다.

NoteBase

선지별 해설

① X

ㄱ, ㄴ, ㄷ, ㄹ은 주기억장치를 캐시메모리보다 빠르게 놓았으므로 틀리다. 올바른 순서는 ㄱ, ㄷ, ㄴ, ㄹ이다.

② O

ㄱ, ㄷ, ㄴ, ㄹ은 레지스터 → 캐시메모리 → 주기억장치 → 하드디스크의 접근 속도 순서와 일치한다.

③ X

ㄱ, ㄷ, ㄹ, ㄴ은 하드디스크를 주기억장치보다 빠르게 놓았으므로 틀리다. 올바른 순서는 ㄱ, ㄷ, ㄴ, ㄹ이다.

④ X

ㄷ, ㄱ, ㄴ, ㄹ은 캐시메모리를 레지스터보다 빠르게 놓았으므로 틀리다. 올바른 순서는 ㄱ, ㄷ, ㄴ, ㄹ이다.

⑤ X

ㄷ, ㄱ, ㄹ, ㄴ은 캐시메모리를 레지스터보다 빠르게, 하드디스크를 주기억장치보다 빠르게 놓았으므로 틀리다. 올바른 순서는 ㄱ, ㄷ, ㄴ, ㄹ이다.

오답 포인트

주기억장치와 캐시메모리의 위치를 바꾸거나, 주기억장치와 하드디스크의 위치를 바꾸는 것이 함정이다. 캐시는 주기억장치보다 CPU에 가깝고 빠르며, 하드디스크는 보조기억장치이므로 가장 느리다.

다시 나오면 이렇게 판단

CPU 내부의 레지스터를 가장 먼저 놓고, CPU와 주기억장치 사이의 캐시메모리, 실행 공간인 주기억장치, 장기 보관용 보조기억장치인 하드디스크의 순서로 판단한다.

암기 포인트

접근 속도는 레지스터 → 캐시 → 주기억장치 → 하드디스크 순이다.

NoteBase

2. 3단계 데이터베이스 구조에서 개체 간의 관계와 제약 조건을 나타내고 데이터베이스의 접근 권한, 보안 및 무결성 규칙에 관한 명세를 정의한 것으로 옳은 것은?

① 외부 스키마 ② 서브 스키마
③ 물리 스키마 ④ 개념 스키마
⑤ 내부 스키마

정답 4번

출제 포인트

3단계 데이터베이스 구조에서 전체 데이터베이스의 논리적 구조와 무결성·보안·접근 권한 규칙을 정의하는 스키마를 구별하는 문제이다.

핵심 개념

외부 스키마(서브 스키마): 개별 사용자나 응용 프로그램이 필요로 하는 데이터의 논리적 모습을 정의한다. 키워드는 사용자별 관점이다.
개념 스키마: 데이터베이스 전체의 통합된 논리적 구조를 정의하며, 개체 간 관계와 제약 조건 및 접근 권한·보안·무결성 규칙을 포함한다. 키워드는 전체 논리 구조이다.
내부 스키마(물리 스키마): 데이터가 저장 장치에 실제로 저장되는 방식과 파일 구조, 인덱스, 레코드 배치 등을 정의한다. 키워드는 물리적 저장 구조이다.
외부 스키마는 사용자별 관점, 개념 스키마는 전체 논리 구조, 내부 스키마는 물리적 저장 구조를 나타낸다.

선지별 해설

① X — 외부 스키마는 개별 사용자나 응용 프로그램이 보는 데이터의 논리적 관점을 정의한다. 틀린 표현은 '외부 스키마'이며, 제시된 명세를 정의하는 것은 '개념 스키마'이다.
② X — 서브 스키마는 외부 스키마와 같은 의미로, 특정 사용자나 응용 프로그램에 필요한 데이터 관점을 정의한다. 틀린 표현은 '서브 스키마'이며, 올바른 표현은 '개념 스키마'이다.
③ X — 물리 스키마는 저장 장치상의 파일 구조, 레코드 배치, 인덱스 등 물리적 저장 방법을 정의한다. 틀린 표현은 '물리 스키마'이며, 올바른 표현은 '개념 스키마'이다.
④ O — 개념 스키마는 데이터베이스 전체의 논리적 구조를 나타내며, 개체 간 관계와 제약 조건, 접근 권한, 보안 및 무결성 규칙을 정의하므로 문제의 설명과 일치한다.
⑤ X — 내부 스키마는 물리 스키마와 같은 수준으로, 데이터의 실제 저장 구조와 접근 경로를 정의한다. 틀린 표현은 '내부 스키마'이며, 올바른 표현은 '개념 스키마'이다.

NoteBase

오답 포인트

‘접근 권한’이라는 말만 보고 사용자 관점인 외부 스키마를 고르기 쉽다. 그러나 문제는 개체 간 관계와 제약 조건을 포함한 데이터베이스 전체의 논리 구조와 보안·무결성 규칙을 함께 묻고 있으므로 개념 스키마에 해당한다. 또한 외부 스키마와 서브 스키마, 내부 스키마와 물리 스키마는 각각 같은 수준을 가리키는 용어이다.

다시 나오면 이렇게 판단

‘사용자별 화면·관점’이면 외부 스키마, ‘전체 데이터의 논리적 관계·제약·무결성’이면 개념 스키마, ‘파일·인덱스·저장 위치’이면 내부 스키마로 판단한다.

암기 포인트

외부는 사용자별, 개념은 전체 논리, 내부는 물리 저장이다. 서브는 외부, 물리는 내부와 대응한다.

NoteBase

3. 다음 중 유닉스 계열의 운영체제에서 사용하는 명령과 그 설명이 옳은 것은?

① grep : 파일에서 특정 단어나 문자열을 포함하는 행을 찾아 출력한다.
② umount : 저장장치의 파일시스템을 디렉터리 트리에 연결한다.
③ ln : 파일을 복사한다.
④ pwd : 사용자의 비밀번호를 설정하거나 변경한다.
⑤ more : 파일에 대한 추가 속성 정보를 출력한다.

정답 1번

출제 포인트

유닉스 계열 운영체제의 기본 명령어와 각 명령어의 실제 기능을 정확히 연결할 수 있는지를 묻는 문제이다.

핵심 개념

grep은 파일이나 입력 내용에서 지정한 문자열 또는 패턴이 들어 있는 행을 검색하여 출력하는 명령어이다. umount는 파일시스템 연결 해제, ln은 링크 생성, pwd는 현재 작업 디렉터리 경로 출력, more는 파일 내용의 화면 단위 출력을 수행한다.

선지별 해설

① O — grep은 파일에서 특정 단어나 문자열 패턴을 포함하는 행을 찾아 출력하므로 설명이 옳다.
② X — 틀린 표현은 '파일시스템을 디렉터리 트리에 연결한다'이다. umount는 연결된 파일시스템을 디렉터리 트리에서 해제하며, 연결하는 명령어는 mount이다.
③ X — 틀린 표현은 '파일을 복사한다'이다. ln은 파일이나 디렉터리에 대한 하드 링크 또는 심볼릭 링크를 생성하며, 파일 복사는 cp가 수행한다.
④ X — 틀린 표현은 '사용자의 비밀번호를 설정하거나 변경한다'이다. pwd는 현재 작업 중인 디렉터리의 경로를 출력하며, 비밀번호 설정·변경에는 passwd를 사용한다.
⑤ X — 틀린 표현은 '파일에 대한 추가 속성 정보를 출력한다'이다. more는 텍스트 파일의 내용을 화면 단위로 나누어 출력하는 명령어이다.

오답 포인트

②는 mount와 umount의 방향을 뒤바꾸었고, ③과 ④는 철자가 비슷하거나 자주 함께 학습하는 cp·passwd의 기능을 바꾸어 제시했다. ⑤도 파일 자체의 정보가 아니라 파일 내용을 읽는 명령어라는 점이 핵심 함정이다.

NoteBase

다시 나오면 이렇게 판단

명령어를 대상과 동작으로 나누어 판단한다. grep은 내용 검색, mount·umount는 파일시스템 연결·해제, ln은 링크 생성, pwd는 현재 경로 출력, more는 내용 열람에 해당한다.

암기 포인트

grep은 행 검색, umount는 연결 해제, ln은 링크, pwd는 현재 경로, more는 화면 단위 열람이다.

NoteBase

4. 컴퓨터 네트워크에서 게이트웨이(gateway)에 대한 설명으로 옳은 것은?

① 디지털 신호와 아날로그 신호 사이의 변환을 담당하는 장치이다.
② 디지털 신호를 멀리 전송할 수 있도록 신호를 증폭하는 역할을 한다.
③ 둘 이상의 LAN을 연결하여 하나의 네트워크로 연결해주는 장치이며, 데이터링크 계층에서만 동작한다.
④ 서로 다른 통신 프로토콜을 사용하는 네트워크 사이를 연결하여 데이터를 교환할 수 있도록 하는 역할을 한다.
⑤ 사람이 읽기 쉬운 도메인 이름을 IP 주소로 변환하는 역할을 한다.

정답 4번

출제 포인트

게이트웨이와 모뎀, 리피터, 브리지, DNS의 기능을 구별하는 문제이다.

핵심 개념

게이트웨이: 서로 다른 통신 프로토콜을 사용하는 네트워크 사이에서 프로토콜을 변환하여 통신이 가능하게 하는 장치이다.
모뎀: 디지털 신호와 아날로그 신호를 서로 변환한다.
리피터: 약해진 신호를 재생·증폭하여 전송 거리를 늘린다.
브리지: 데이터링크 계층에서 둘 이상의 LAN 세그먼트를 연결한다.
DNS: 사람이 읽기 쉬운 도메인 이름을 IP 주소로 변환한다.

선지별 해설

① X — '디지털 신호와 아날로그 신호 사이의 변환'은 게이트웨이가 아니라 모뎀의 기능이다. 올바르게는 '모뎀은 디지털 신호와 아날로그 신호를 서로 변환한다'이다.
② X — '신호를 증폭하여 멀리 전송'하는 것은 게이트웨이가 아니라 리피터의 기능이다. 올바르게는 '리피터는 약해진 신호를 재생·증폭하여 전송 거리를 연장한다'이다.
③ X — '둘 이상의 LAN을 연결하며 데이터링크 계층에서 동작'하는 것은 게이트웨이가 아니라 브리지의 기능이다. 올바르게는 '브리지는 데이터링크 계층에서 LAN 세그먼트를 연결한다'이다.
④ O — 게이트웨이는 서로 다른 통신 프로토콜을 사용하는 네트워크 사이를 연결하고 프로토콜을 변환하여 데이터를 교환할 수 있게 한다.
⑤ X — '도메인 이름을 IP 주소로 변환'하는 것은 게이트웨이가 아니라 DNS의 기능이다. 올바르게는 'DNS는 도메인 이름을 IP 주소로 변환한다'이다.

NoteBase

오답 포인트

네트워크 장치가 모두 '연결'이나 '전송'을 돕는다는 공통점 때문에 기능을 혼동하기 쉽다. 특히 ③의 LAN 연결은 브리지, ④의 서로 다른 프로토콜 간 연결은 게이트웨이라는 차이가 핵심이다.

다시 나오면 이렇게 판단

선지의 핵심 동사를 기준으로 판단한다. 신호 변환은 모뎀, 신호 재생·증폭은 리피터, LAN 세그먼트 연결은 브리지, 서로 다른 프로토콜의 변환은 게이트웨이, 이름을 IP 주소로 변환하는 것은 DNS이다.

암기 포인트

게이트웨이는 서로 다른 프로토콜 사이의 관문이다.

NoteBase

5. 이진트리를 전위순회(preorder)와 중위순회(inorder)로 방문한 결과가 다음과 같다. 이 이진트리를 후위순회(postorder)로 방문한 결과는 무엇인가?

전위순회 = [ABCDEFG]
중위순회 = [CBDAFGE]

① [GFEDCBA]
② [FEGDCBA]
③ [CBDGFEA]
④ [CDBGFEA]
⑤ [CDBFGEA]

정답 4번

출제 포인트

전위순회와 중위순회 결과로 이진트리의 구조를 복원한 뒤, 왼쪽 서브트리 → 오른쪽 서브트리 → 루트 순서인 후위순회 결과를 구하는 문제이다.

올바른 순서

1. 전위순회의 첫 원소 A를 전체 루트로 정한다. 중위순회에서 A의 왼쪽 [CBD]는 왼쪽 서브트리, 오른쪽 [FGE]는 오른쪽 서브트리이다.
2. 왼쪽 서브트리의 전위순회 [BCD]에서 B가 루트이다. 중위순회 [CBD]를 기준으로 C는 B의 왼쪽 자식, D는 B의 오른쪽 자식이므로 후위순회는 [CDB]이다.
3. 오른쪽 서브트리의 전위순회 [EFG]에서 E가 루트이다. 중위순회 [FGE]에서 E의 왼쪽에 F와 G가 있고, 그 부분의 루트 F는 오른쪽 자식 G를 가지므로 후위순회는 [GFE]이다.
4. 왼쪽 서브트리 [CDB], 오른쪽 서브트리 [GFE], 전체 루트 A를 차례로 연결하면 [CDBGFEA]이다.

왜 이 순서: 후위순회에서는 각 서브트리의 자식들을 모두 방문한 뒤 그 서브트리의 루트를 방문하므로, C와 D가 B보다 앞서고 G와 F가 E보다 앞서며 전체 루트 A가 마지막에 온다.

NoteBase

선지별 해설

① X — [GFEDCBA]는 전체 루트 A가 마지막이라는 점만 맞고 서브트리 순서가 틀리다. 특히 왼쪽 서브트리의 후위순회는 [DCB]가 아니라 [CDB]이다. 올바르게는 [CDBGFEA]이다.
② X — [FEGDCBA]는 E의 왼쪽 서브트리에서 G가 F의 오른쪽 자식인데도 F를 G보다 먼저 방문하고, 전체의 오른쪽 서브트리를 왼쪽 서브트리보다 먼저 배치했다. 올바르게는 [CDBGFEA]이다.
③ X — [CBDGFEA]에서 왼쪽 서브트리 [CBD]가 틀리다. B는 C와 D의 부모이므로 후위순회에서는 B가 두 자식 뒤에 와서 [CDB]가 되어야 한다. 올바르게는 [CDBGFEA]이다.
④ O — 왼쪽 서브트리의 후위순회 [CDB], 오른쪽 서브트리의 후위순회 [GFE], 루트 A를 연결한 [CDBGFEA]와 일치한다.
⑤ X — [CDBFGEA]에서 오른쪽 서브트리 [FGE]가 틀리다. F의 오른쪽 자식이 G이므로 후위순회에서는 G를 먼저 방문한 뒤 F와 E를 방문하여 [GFE]가 되어야 한다. 올바르게는 [CDBGFEA]이다.

오답 포인트

중위순회 [FGE]를 그대로 후위순회에 옮기면 ⑤를 고르기 쉽다. 중위순회는 왼쪽 → 루트 → 오른쪽이고 후위순회는 왼쪽 → 오른쪽 → 루트이므로, F의 오른쪽 자식 G를 F보다 먼저 기록해야 한다.

다시 나오면 이렇게 판단

전위순회의 첫 원소로 루트를 정하고, 그 루트를 기준으로 중위순회를 왼쪽과 오른쪽 구간으로 나눈다. 이 과정을 각 구간에 반복해 트리를 복원한 뒤, 각 노드를 왼쪽 → 오른쪽 → 루트 순서로 기록한다.

암기 포인트

전위는 루트가 먼저, 중위는 루트가 가운데, 후위는 루트가 마지막이다.

NoteBase

6. 다음 중 CSS(Cascading Style Sheets)에 대한 설명으로 옳은 것은?

① HTML 문서의 내용을 웹 브라우저에 어떻게 표현할 것인가를 기술하는 역할을 한다.
② 웹페이지의 동작을 정의하는 스크립트 프로그래밍 언어이다.
③ CSS 스타일은 반드시 HTML 문서 내에 정의해야 한다.
④ CSS의 주석은 '<!--'와 '-->' 사이에 작성한다.
⑤ 의사 클래스(pseudo-class) : focus는 사용자가 특정 요소 위로 마우스 포인터를 옮겨놓으면 적용된다.

정답 1번

출제 포인트

CSS의 역할과 작성 위치, 주석 문법, 의사 클래스의 의미를 구별하는 문제이다.

핵심 개념

CSS(Cascading Style Sheets)는 HTML로 구성한 웹 문서의 색상, 글꼴, 크기, 배치 등 표현과 스타일을 지정하는 언어이다.
HTML은 문서의 구조와 내용을 담당하고, CSS는 표현을 담당하며, JavaScript는 동작과 상호작용을 담당한다.
CSS는 인라인 스타일, HTML 문서 내부의 style 요소, 별도의 외부 CSS 파일 방식으로 적용할 수 있다.
CSS 주석은 /* 내용 */ 형식이며, :focus는 요소가 입력 초점을 얻은 상태를 선택한다.

선지별 해설

① O — CSS는 HTML 문서의 내용을 웹 브라우저에 어떤 모양으로 표현할지를 지정한다. 색상, 글꼴, 여백, 배치 등을 정하는 것이 대표적인 역할이다.
② X — 틀린 표현은 '웹페이지의 동작을 정의하는 스크립트 프로그래밍 언어'이다. 올바르게는 JavaScript가 웹페이지의 동작과 상호작용을 정의하며, CSS는 문서의 표현과 스타일을 지정한다.
③ X — 틀린 표현은 '반드시 HTML 문서 내에 정의해야 한다'이다. 올바르게는 CSS를 HTML 요소의 style 속성에 직접 작성하거나 HTML 문서의 style 요소에 작성할 수 있고, 별도의 외부 CSS 파일로 분리하여 연결할 수도 있다.
④ X — 틀린 표현은 '<!--'와 '-->' 사이에 작성한다는 부분이다. 올바르게는 CSS 주석을 /*와 */ 사이에 작성하며, <!--와 -->는 HTML 주석 문법이다.
⑤ X — 틀린 표현은 ':focus는 사용자가 특정 요소 위로 마우스 포인터를 옮겨놓으면 적용된다'이다. 올바르게는 :focus가 키보드 이동이나 클릭 등으로 입력 초점을 얻은 요소에 적용되고, 마우스 포인터를 올린 상태에는 :hover가 적용된다.

NoteBase

오답 포인트

CSS, HTML, JavaScript의 역할을 서로 바꾸거나 :focus와 :hover를 혼동하기 쉽다. 특히 ⑤의 '마우스 포인터를 옮겨놓으면'은 :focus가 아니라 :hover를 가리키는 표현이다.

다시 나오면 이렇게 판단

문서의 구조·내용은 HTML, 표현·꾸밈은 CSS, 동작·상호작용은 JavaScript로 구분한다. CSS 문법에서는 주석은 /* */, 초점 상태는 :focus, 마우스를 올린 상태는 :hover로 판단한다.

암기 포인트

HTML은 구조, CSS는 표현, JavaScript는 동작이다. CSS 주석은 /* */, 초점은 :focus, 마우스 올림은 :hover이다.

NoteBase

7. 다음 C++ 프로그램의 실행결과로 옳은 것은?

```
#include <iostream>
using namespace std;

int main()
{
    int x;
    for (x = 1; x <= 7; x++) {
        if (x == 5)
            continue;
        else if (x == 6)
            break;
        cout << x;
    }
    return 0;
}
```

① 123
② 1234
③ 12345
④ 12346
⑤ 12347

정답 2번

출제 포인트

이 문제는 for문의 반복 흐름에서 continue와 break가 cout의 실행 여부에 어떤 영향을 주는지 묻는 문제이다. continue는 현재 반복의 남은 문장을 건너뛰고 다음 반복으로 넘어가며, break는 반복문 자체를 즉시 종료한다.

NoteBase

올바른 순서

1. x가 1, 2, 3, 4일 때는 두 조건이 모두 거짓이므로 각각 cout이 실행되어 1234가 출력된다.
2. x가 5일 때는 continue가 실행되어 그 아래의 cout을 건너뛰므로 5는 출력되지 않는다.
3. 증감식 x++에 따라 x가 6이 되어 다음 반복을 시작한다.
4. x가 6일 때는 break가 실행되어 cout을 거치지 않고 for문이 종료되므로 6과 7은 출력되지 않는다.

왜 이 순서: continue는 현재 반복만 건너뛰어 다음 반복으로 이어지지만, break는 반복문 전체를 끝내므로 x가 6에 도달한 뒤에는 더 이상 출력이 발생하지 않는다.

선지별 해설

① X — 123에서 끝나지 않는다. x가 4일 때도 조건에 걸리지 않아 4가 출력되므로 올바르게는 1234이다.
② O — x가 1부터 4까지는 출력되고, x=5에서는 continue로 출력이 생략되며, x=6에서는 break로 반복문이 종료되므로 실행 결과는 1234이다.
③ X — 12345에서 5가 잘못되었다. x=5일 때 continue가 cout을 건너뛰므로 올바르게는 1234이다.
④ X — 12346에서 6이 잘못되었다. x=6일 때 break가 cout보다 먼저 실행되므로 6은 출력되지 않으며, 올바르게는 1234이다.
⑤ X — 12347에서 7이 잘못되었다. x=6에서 break로 for문 전체가 종료되어 x=7의 반복은 시작되지 않으며, 올바르게는 1234이다.

오답 포인트

continue와 break를 모두 단순히 해당 값만 건너뛰는 명령으로 혼동하기 쉽다. continue는 x=5의 cout만 생략한 뒤 증감식과 다음 반복으로 진행하지만, break는 x=6에서 반복문 전체를 종료하므로 6뿐 아니라 이후의 7도 출력되지 않는다.

다시 나오면 이렇게 판단

반복문의 각 회차마다 조건문이 출력문보다 먼저 실행되는지 확인한다. continue를 만나면 그 회차의 남은 문장을 건너뛰고 다음 회차로 이동하며, break를 만나면 이후 회차까지 모두 중단되는 것으로 판단한다.

암기 포인트

continue는 이번 회차만 건너뛰고, break는 반복문 전체를 끝낸다. 따라서 5는 건너뛰고 6에서 종료되어 1234가 남는다.

NoteBase

8. 다음의 설명 중 옳지 않은 것은?

① 지도 학습(supervised learning)은 입력과 출력 사이의 매핑을 학습하는 것이며, 입력과 출력 쌍이 데이터로 주어지는 경우에 적용한다.

② 지도 학습 종류에는 예측변수의 특성을 사용해 목표 수치를 예측하는 회귀(regression) 기법도 있다.

③ 강화 학습(reinforcement learning)은 주어진 입력에 대한 일련의 행동 결과에 대해 보상(reward)이 주어지게 되며, 시스템은 이러한 보상을 이용해 학습을 행한다.

④ 비지도 학습(unsupervised learning)은 입력만 있고 출력은 없는 경우에 적용하며, 입력 사이의 규칙성 등을 찾아내는 게 목표이다.

⑤ 어떤 종류의 값을 군집(clustering)하고 이를 분류(classification)하는 기법은 비지도 학습의 한 종류이다.

정답 5번

출제 포인트

지도 학습·비지도 학습·강화 학습의 학습 방식과, 회귀·분류·군집화가 어느 방식에 속하는지를 구분하는 문제이다.

핵심 개념

지도 학습: 정답 역할을 하는 출력값(레이블)이 포함된 입력-출력 쌍으로 입력과 출력의 관계를 학습한다. 대표 과제는 연속적인 수치를 예측하는 회귀와 미리 정해진 범주를 예측하는 분류이다.
비지도 학습: 출력 레이블 없이 입력 데이터에 내재한 구조나 규칙을 찾는다. 대표 과제는 유사한 데이터끼리 묶는 군집화이다.
강화 학습: 에이전트가 환경에서 행동하고 받은 보상을 바탕으로 누적 보상을 크게 만드는 행동 전략을 학습한다.

선지별 해설

① O — 지도 학습은 입력과 정답 출력의 쌍을 학습 데이터로 사용하여 둘 사이의 매핑을 학습한다.
② O — 회귀는 입력 특성을 바탕으로 가격이나 온도와 같은 연속적인 목표 수치를 예측하는 지도 학습 기법이다.
③ O — 강화 학습에서는 행동 결과에 보상이 주어지고, 에이전트는 이 보상을 이용해 더 나은 행동 전략을 학습한다.
④ O — 비지도 학습은 정답 출력 없이 입력만으로 데이터의 숨은 구조, 규칙성, 유사성 등을 찾는 것을 목표로 한다.
⑤ X — 틀린 부분은 '군집하고 이를 분류하는 기법은 비지도 학습'이라는 표현이다. 군집화는 레이블 없이 유사한 데이터를 묶는 비지도 학습이지만, 분류는 레이블이 있는 데이터로 미리 정해진 범주를 예측하는 지도 학습이다. 올바르게는 '유사한 데이터들을 군집화하는 기법은 비지도 학습의 한 종류이다'라고 해야 한다.

NoteBase

오답 포인트

⑤는 일상적으로 '나눈다'는 뜻이 비슷한 군집화와 분류를 한데 묶어 혼동하게 한다. 군집화는 정답 범주가 없는 상태에서 집단을 발견하고, 분류는 이미 정해진 정답 범주를 예측한다는 차이가 핵심이다.

다시 나오면 이렇게 판단

먼저 정답 레이블의 존재 여부를 확인한다. 레이블로 연속값을 예측하면 회귀, 레이블로 범주를 예측하면 분류, 레이블 없이 유사성에 따라 묶으면 군집화, 행동과 보상으로 전략을 익히면 강화 학습으로 판단한다.

암기 포인트

회귀·분류는 지도 학습, 군집화는 비지도 학습, 행동·보상은 강화 학습이다.

NoteBase

9. 다음 중 RAID의 레벨에 대한 설명으로 옳은 것은?

① 레이드 레벨 0은 동일하게 미러링된 디스크 세트로 구성된다.

② 레이드 레벨 1은 데이터를 2개 이상의 디스크에 블록 단위로 스트라이핑하여 저장하며, 다른 레이드 레벨에 비해 오류에 취약하다.

③ 레이드 레벨 3은 데이터를 다수의 디스크에 스트라이핑하여 저장하며, 하나의 드라이브에 패리티를 저장한다.

④ 레이드 레벨 5는 데이터와 패리티를 비트단위로 다수의 디스크에 스트라이핑하여 저장한다.

⑤ 레이드 레벨 6은 데이터를 2개 이상의 디스크에 블록 단위로 스트라이핑하여 저장하며, 패리티는 포함하지 않는다.

정답 3번

출제 포인트

RAID 레벨별 데이터 분산 단위, 미러링 방식, 패리티의 사용 여부와 저장 위치를 구별하는 문제이다.

핵심 개념

RAID 0은 데이터를 블록 단위로 스트라이핑하며 중복 정보가 없다. RAID 1은 같은 데이터를 복제하여 저장하는 미러링 방식이다. RAID 3은 데이터를 바이트 단위로 스트라이핑하고 전용 패리티 디스크 한 대를 사용한다. RAID 5는 블록 단위 스트라이핑과 분산 패리티를 사용한다. RAID 6은 블록 단위 스트라이핑과 이중 분산 패리티를 사용한다.

NoteBase

선지별 해설

① X
틀린 표현은 '동일하게 미러링된 디스크 세트'이다. RAID 0은 미러링이 아니라 데이터를 여러 디스크에 블록 단위로 나누어 기록하는 스트라이핑 방식이며 패리티도 사용하지 않는다.

② X
틀린 표현은 '블록 단위로 스트라이핑'과 '오류에 취약하다'이다. RAID 1은 데이터를 스트라이핑하지 않고 두 개 이상의 디스크에 동일하게 복제하는 미러링 방식이므로 디스크 장애에 대한 내결함성을 제공한다.

③ O
RAID 3은 데이터를 다수의 디스크에 바이트 단위로 스트라이핑하고, 별도의 한 디스크에 전용 패리티를 저장한다. 따라서 제시된 설명이 옳다.

④ X
틀린 표현은 '비트 단위'이다. RAID 5는 데이터를 블록 단위로 스트라이핑하며 패리티를 특정 한 디스크에만 두지 않고 여러 디스크에 분산하여 저장한다.

⑤ X
틀린 표현은 '패리티는 포함하지 않는다'이다. RAID 6은 데이터를 블록 단위로 스트라이핑하고 서로 독립적인 이중 패리티를 여러 디스크에 분산하여 저장한다.

오답 포인트

RAID 0과 RAID 1의 스트라이핑·미러링을 서로 바꾸거나, RAID 3·5·6의 스트라이핑 단위와 패리티 위치를 혼동하기 쉽다. 특히 ④의 '비트 단위'와 ⑤의 '패리티는 포함하지 않는다'가 핵심 함정이다.

다시 나오면 이렇게 판단

각 레벨을 '데이터 분산 단위 — 패리티 위치 — 패리티 개수'의 순서로 판별한다. RAID 0은 블록·패리티 없음, RAID 1은 미러링, RAID 3은 바이트·전용 패리티, RAID 5는 블록·단일 분산 패리티, RAID 6은 블록·이중 분산 패리티이다.

암기 포인트

0은 블록 스트라이핑, 1은 미러링, 3은 바이트 + 전용 패리티, 5는 블록 + 분산 패리티, 6은 블록 + 이중 분산 패리티이다.

NoteBase

10. 관계 데이터 모델의 설명으로 옳지 않은 것은?

① 릴레이션(relation)의 튜플(tuple)들은 모두 상이하다.
② 릴레이션에서 속성(attribute)들 간의 순서는 의미가 없다.
③ 한 릴레이션에 포함된 튜플 사이에는 순서가 없다.
④ 튜플은 원자값으로 분해가 불가능하다.
⑤ 릴레이션은 속성들을 가지고 있는 테이블이다.

정답 4번

출제 포인트

관계 데이터 모델에서 릴레이션, 튜플, 속성이 갖는 기본 성질을 구별하는 문제이다. 특히 원자성은 튜플 전체가 아니라 각 속성값에 적용된다는 점이 핵심이다.

핵심 개념

릴레이션: 행과 열로 구성된 테이블이며, 속성들의 집합과 튜플들의 집합으로 표현된다.
튜플: 릴레이션의 한 행으로, 동일한 튜플은 중복되지 않고 튜플 사이의 순서에도 의미가 없다.
속성: 릴레이션의 한 열로, 속성 사이의 순서에는 의미가 없으며 각 속성값은 더 나눌 수 없는 원자값이어야 한다.

선지별 해설

① O — 릴레이션은 튜플의 집합이므로 동일한 튜플이 중복될 수 없다. 따라서 릴레이션의 튜플들은 서로 상이하다.
② O — 릴레이션에서 속성은 이름으로 식별하므로 속성을 나열한 물리적 순서에는 의미가 없다.
③ O — 릴레이션은 튜플의 집합이므로 튜플 사이의 순서에는 의미가 없다.
④ X — 틀린 표현은 '튜플은 원자값으로 분해가 불가능하다'이다. 올바르게는 '튜플을 구성하는 각 속성값은 더 이상 분해할 수 없는 원자값이다'이다. 튜플 자체는 여러 속성값으로 구성된다.
⑤ O — 릴레이션은 속성을 열로, 튜플을 행으로 가지는 테이블 형태로 표현된다.

오답 포인트

④는 '속성값의 원자성'을 '튜플의 원자성'으로 바꾸어 놓았다. 튜플은 여러 속성값으로 나눌 수 있지만, 하나의 속성에 저장되는 값은 원자값이어야 한다.

다시 나오면 이렇게 판단

관계 모델의 성질을 판단할 때 집합의 성질과 원자성의 대상을 나누어 본다. 속성과 튜플의 순서는 의미가 없고 튜플은 중복되지 않으며, 원자성은 튜플 전체가 아니라 각 속성값에 적용된다.

NoteBase

암기 포인트

순서 없음, 튜플 중복 없음, 속성값은 원자값이다.

NoteBase

11. 다음 레지스터 중에서 기억장치를 출입하는 데이터를 잠시 저장하는 용도로 사용되며, CPU에 의해 명령어가 처리되기 위해서 반드시 거쳐야 하는 레지스터로 옳은 것은?

① Status Register
② Program Status Word Register
③ Memory Address Register
④ Memory Buffer Register
⑤ Index Register

정답 4번

출제 포인트

이 문제는 CPU의 주요 레지스터 가운데 기억장치와 주고받는 데이터나 명령어를 임시 저장하는 레지스터를 구별하는 문제이다.

핵심 개념

Memory Buffer Register(MBR)는 CPU와 기억장치 사이에서 전송되는 데이터나 명령어를 일시적으로 보관하는 레지스터이다. Memory Address Register(MAR)는 접근할 기억장치의 주소를 저장한다. 즉 MAR는 '어디에 접근하는가'를, MBR은 '무엇을 주고받는가'를 담당한다.

선지별 해설

① X — Status Register는 연산 결과에 따른 부호, 영, 자리올림, 오버플로 등의 상태 플래그를 저장한다. 틀린 표현은 기억장치 출입 데이터의 임시 저장 용도로 본 점이며, 올바르게는 CPU의 연산 상태를 나타내는 정보를 저장하는 레지스터이다.
② X — Program Status Word Register는 조건 코드와 제어 상태 등 프로그램 실행 상태 정보를 보관한다. 틀린 표현은 기억장치와 전송되는 데이터의 임시 저장 용도로 본 점이며, 올바르게는 CPU의 현재 실행 상태를 나타내는 정보를 저장하는 레지스터이다.
③ X — Memory Address Register는 CPU가 읽거나 쓸 기억장치 위치의 주소를 저장한다. 틀린 표현은 데이터 자체를 저장한다고 본 점이며, 올바르게는 기억장치에 접근할 주소를 저장하는 레지스터이다.
④ O — Memory Buffer Register는 기억장치에서 CPU로 들어오거나 CPU에서 기억장치로 나가는 데이터 또는 명령어를 잠시 저장하며, 기억장치와 CPU 사이의 전송 과정에서 거치는 레지스터이다.
⑤ X — Index Register는 인덱스 주소 지정에서 기준 주소에 더할 변위나 인덱스 값을 저장한다. 틀린 표현은 기억장치 출입 데이터의 임시 저장 용도로 본 점이며, 올바르게는 유효주소 계산에 사용하는 인덱스 값을 저장하는 레지스터이다.

NoteBase

오답 포인트

③의 Memory Address Register와 ④의 Memory Buffer Register를 혼동하기 쉽다. MAR에는 주소가 들어가고, MBR에는 그 주소를 통해 실제로 주고받는 데이터나 명령어가 들어간다.

다시 나오면 이렇게 판단

문장에서 '기억장치의 주소'를 묻는다면 MAR, '기억장치와 주고받는 데이터나 명령어를 임시 저장'을 묻는다면 MBR로 판단한다.

암기 포인트

MAR는 Address, MBR는 Buffer이다. MAR는 주소, MBR는 전송 내용이다.

NoteBase

12. 다음 <보기> 중 트랜잭션의 ACID 특성에 해당되는 것을 나열한 것으로 옳은 것은?

<보 기>

ㄱ. 하나의 트랜잭션은 부분적으로 반영될 수 있다.

ㄴ. 완료된 트랜잭션의 결과는 시스템의 장애 등이 발생해도 손실되지 않는다.

ㄷ. 트랜잭션 실행이 완료되면 언제나 일관성 있는 데이터베이스 상태로 유지되어야 한다.

ㄹ. 트랜잭션이 갱신 중인 데이터에 다른 트랜잭션이 접근할 수 있다.

① ㄱ, ㄴ ② ㄱ, ㄷ

③ ㄴ, ㄷ ④ ㄴ, ㄹ

⑤ ㄷ, ㄹ

정답 3번

출제 포인트

트랜잭션의 ACID 특성인 원자성, 일관성, 고립성, 지속성의 의미를 구별하는 문제이다.

핵심 개념

원자성(Atomicity): 트랜잭션의 연산은 전부 반영되거나 전혀 반영되지 않아야 한다.

일관성(Consistency): 트랜잭션이 성공적으로 끝나면 데이터베이스는 정의된 무결성 제약을 만족하는 일관된 상태를 유지해야 한다.

고립성(Isolation): 실행 중인 트랜잭션의 중간 결과에 다른 트랜잭션이 접근하거나 간섭할 수 없어야 한다.

지속성(Durability): 완료된 트랜잭션의 결과는 시스템 장애가 발생해도 영구적으로 보존되어야 한다.

따라서 ㄱ은 원자성에 어긋나고, ㄴ은 지속성, ㄷ은 일관성에 해당하며, ㄹ은 고립성에 어긋난다.

NoteBase

선지별 해설

① X — ㄱ은 하나의 트랜잭션이 부분적으로 반영될 수 있다고 하므로 원자성에 어긋난다. 틀린 표현은 '부분적으로 반영될 수 있다'이며, 올바르게는 '트랜잭션의 모든 연산이 전부 반영되거나 전혀 반영되지 않아야 한다'이다. ㄴ은 지속성에 해당하지만 ㄱ이 틀렸으므로 옳은 조합이 아니다.
② X — ㄷ은 일관성에 해당하지만 ㄱ이 원자성에 어긋난다. 틀린 ㄱ은 '하나의 트랜잭션은 부분적으로 반영될 수 없다'로 고쳐야 한다.
③ O — ㄴ은 완료된 트랜잭션의 결과가 장애 후에도 보존된다는 지속성을 설명하고, ㄷ은 완료 후 데이터베이스가 일관된 상태를 유지한다는 일관성을 설명한다. 두 설명이 모두 옳다.
④ X — ㄴ은 지속성에 해당하지만 ㄹ은 갱신 중인 데이터에 다른 트랜잭션이 접근할 수 있다고 하므로 고립성에 어긋난다. 틀린 ㄹ은 '트랜잭션이 갱신 중인 데이터에는 완료 전까지 다른 트랜잭션이 접근하거나 간섭할 수 없다'로 고쳐야 한다.
⑤ X — ㄷ은 일관성에 해당하지만 ㄹ이 고립성에 어긋난다. 틀린 ㄹ은 '실행 중인 트랜잭션의 중간 결과는 다른 트랜잭션으로부터 격리되어야 한다'로 고쳐야 한다.

오답 포인트

ㄱ의 '부분적으로 반영'과 ㄹ의 '다른 트랜잭션이 접근'은 각각 원자성과 고립성을 설명하는 것처럼 보이지만, 두 문장 모두 ACID가 보장해야 할 내용을 반대로 서술한 함정이다. 또한 ㄴ의 장애 후 보존은 일관성이 아니라 지속성이다.

다시 나오면 이렇게 판단

'전부 또는 전혀 없음'은 원자성, '정상 완료 후 무결성 유지'는 일관성, '동시 실행 간 중간 결과 차단'은 고립성, '완료 결과의 영구 보존'은 지속성으로 판단한다. 문장에 '부분 반영 가능'이나 '갱신 중 접근 가능'이 나오면 각각 원자성과 고립성에 어긋나는지 확인한다.

암기 포인트

ACID는 전부 아니면 전무(A), 일관된 상태(C), 서로 격리(I), 완료 결과 영구 보존(D)이다.

NoteBase

13. 다음 <보기> 중 프로세스 스케줄링을 선점 스케줄링과 비선점 스케줄링으로 구분한 것으로 옳은 것은?

<보 기>

ㄱ. RR(Round Robin)
ㄴ. SJF(Shortest Job First)
ㄷ. SRT(Shortest Remaining Time)
ㄹ. HRN(Highest Response ratio Next)
ㅁ. MFQ(Multilevel Feedback Queue)
ㅂ. MLQ(MultiLevel Queue)

	선점 스케줄링	비선점 스케줄링
①	ㅁ, ㅂ	ㄱ, ㄴ, ㄷ, ㄹ
②	ㄱ, ㄴ, ㄷ	ㄹ, ㅁ, ㅂ
③	ㄱ, ㅁ, ㅂ	ㄴ, ㄷ, ㄹ
④	ㄱ, ㄴ, ㄷ, ㄹ	ㅁ, ㅂ
⑤	ㄱ, ㄷ, ㅁ, ㅂ	ㄴ, ㄹ

정답 5번

출제 포인트

프로세스가 실행 중 강제로 CPU를 빼앗길 수 있는 선점 스케줄링과, 실행 중인 프로세스가 종료되거나 대기 상태가 될 때까지 CPU를 유지하는 비선점 스케줄링을 구분하는 문제이다.

핵심 개념

선점 스케줄링: 운영체제가 실행 중인 프로세스의 CPU 사용을 중단시키고 다른 프로세스에 CPU를 배정할 수 있는 방식이다. RR은 시간 할당량 만료, SRT는 더 짧은 잔여 시간의 프로세스 도착, MFQ는 동적으로 변하는 다단계 피드백 큐, MLQ는 큐 사이의 우선순위에 따라 선점이 가능하므로 선점 방식으로 분류한다.
비선점 스케줄링: 실행 중인 프로세스가 종료되거나 스스로 대기 상태가 될 때까지 CPU를 빼앗지 않는 방식이다. SJF와 HRN이 이에 해당한다.
핵심 대비: SJF의 선점형 변형이 SRT이다.

NoteBase

선지별 해설

① X — MFQ와 MLQ만 선점으로 둔 부분이 틀렸다. RR과 SRT도 실행 중인 프로세스를 선점할 수 있다. 올바른 분류는 선점이 RR·SRT·MFQ·MLQ이고, 비선점이 SJF·HRN이다.
② X — SJF를 선점으로, MFQ와 MLQ를 비선점으로 분류한 부분이 틀렸다. SJF는 비선점이고 MFQ와 MLQ는 선점으로 분류한다. 올바른 분류는 선점이 RR·SRT·MFQ·MLQ이고, 비선점이 SJF·HRN이다.
③ X — SJF와 HRN을 선점으로, SRT를 비선점으로 분류한 부분이 틀렸다. SJF와 HRN은 비선점이며, SRT는 새로 도착한 프로세스의 남은 시간이 더 짧으면 현재 프로세스를 중단시키므로 선점이다. 올바른 분류는 선점이 RR·SRT·MFQ·MLQ이고, 비선점이 SJF·HRN이다.
④ X — SJF와 HRN을 선점으로, MFQ와 MLQ를 비선점으로 분류한 부분이 틀렸다. SJF와 HRN은 비선점이고 MFQ와 MLQ는 선점으로 분류한다. 올바른 분류는 선점이 RR·SRT·MFQ·MLQ이고, 비선점이 SJF·HRN이다.
⑤ O — RR·SRT·MFQ·MLQ를 선점 스케줄링으로, SJF·HRN을 비선점 스케줄링으로 정확히 구분하였다.

오답 포인트

SJF와 SRT의 이름이 비슷하다는 점이 핵심 함정이다. SJF는 선택된 작업이 끝날 때까지 실행하는 비선점 방식이고, SRT는 남은 시간이 더 짧은 작업이 도착하면 현재 작업을 중단하는 선점 방식이다. 또한 MFQ는 프로세스의 큐 이동이 가능하지만 MLQ는 큐가 고정된다는 차이가 있으나, 이 문항의 분류에서는 둘 다 선점 방식에 포함된다.

다시 나오면 이렇게 판단

시간 할당량이 끝나거나 더 높은 우선순위·더 짧은 잔여 시간의 프로세스가 나타났을 때 실행 중인 프로세스를 중단할 수 있는지를 확인한다. 중단할 수 있으면 선점, 종료 또는 대기 상태가 될 때까지 기다리면 비선점이다.

암기 포인트

선점은 RR·SRT·MFQ·MLQ, 비선점은 SJF·HRN이다. SJF의 선점형이 SRT이다.

NoteBase

14. 다음 중 TCP, IP, UDP에 대한 설명으로 옳지 않은 것은?

① TCP는 순서제어, 에러제어 및 흐름제어, 패킷 다중화 기능을 제공한다.
② IP는 패킷의 분해, 조립, 출발지와 목적지의 포트번호 지정 기능을 제공한다.
③ UDP는 TCP와 함께 전송 계층에 속하나 TCP와 다르게 전송확인이나 신뢰성에 대한 고려는 없다.
④ IP는 에러 감지 및 복구 기능이 없으므로 신뢰도가 낮다.
⑤ UDP는 데이터그램 단위의 고속 전송을 할 때 사용한다.

정답 2번

출제 포인트

이 문제는 TCP·UDP의 전송 계층 기능과 IP의 인터넷 계층 기능을 구분하는 문제이다.

핵심 개념

TCP: 연결 지향형 전송 프로토콜로, 순서 제어·오류 제어·흐름 제어를 통해 신뢰성 있는 전송을 제공하고 포트번호로 응용 프로세스를 구분한다.
UDP: 비연결형 전송 프로토콜로, 신뢰성 보장 기능을 최소화하여 데이터그램을 빠르게 전송하고 포트번호로 응용 프로세스를 구분한다.
IP: 인터넷 계층 프로토콜로, 출발지와 목적지의 IP 주소를 이용해 패킷을 전달하며 필요할 때 단편화와 재조립을 수행한다.
구별 기준: IP 주소는 호스트를, 포트번호는 호스트 내부의 응용 프로세스를 식별한다.

선지별 해설

① O — TCP는 세그먼트의 순서를 맞추는 순서 제어, 오류 발생 시 재전송하는 오류 제어, 수신 측 처리 능력에 맞추는 흐름 제어를 제공한다. 또한 포트번호를 이용하여 여러 응용 프로세스의 데이터를 다중화하고 역다중화한다.
② X — 틀린 부분은 'IP가 출발지와 목적지의 포트번호를 지정한다'는 표현이다. 포트번호는 TCP 또는 UDP 헤더에 포함되며 전송 계층에서 응용 프로세스를 식별한다. 올바르게는 IP가 출발지와 목적지의 IP 주소를 지정한다고 해야 한다.
③ O — UDP는 TCP와 같은 전송 계층 프로토콜이지만 연결 설정, 순서 보장, 재전송과 같은 전송 확인 및 신뢰성 보장 기능을 제공하지 않는다.
④ O — IP는 최선형 방식으로 패킷을 전달할 뿐 종단 간 오류 복구나 재전송을 보장하지 않으므로 TCP보다 신뢰성이 낮다.
⑤ O — UDP는 연결 설정과 재전송 절차가 없어 오버헤드가 작으므로 데이터그램 단위의 고속 전송에 사용된다.

NoteBase

오답 포인트

②는 IP 주소와 포트번호의 역할을 바꾸어 놓은 선지이다. '출발지·목적지'라는 표현은 두 헤더에 모두 등장할 수 있지만, IP 헤더에는 IP 주소가 있고 TCP·UDP 헤더에는 포트번호가 있다.

다시 나오면 이렇게 판단

주소의 대상을 확인한다. 네트워크상의 호스트를 찾는 주소이면 IP의 IP 주소이고, 호스트 안에서 통신할 응용 프로그램을 찾는 번호이면 TCP·UDP의 포트번호이다.

암기 포인트

IP는 호스트 주소와 패킷 전달, TCP·UDP는 포트번호와 프로세스 전달이다. TCP는 신뢰성, UDP는 신속성이다.

NoteBase

15. 다음 중 정렬 알고리즘에 대한 설명으로 옳은 것은?

① 선택 정렬과 힙 정렬 알고리즘의 시간 복잡도는 $O(n\log n)$이다.
② 퀵 정렬과 합병 정렬은 분할정복 방법을 사용하는 알고리즘이다.
③ 힙 정렬과 삽입 정렬은 안정적 정렬 알고리즘이다.
④ 최악의 경우 퀵 정렬의 성능은 $O(n\log n)$이다.
⑤ 선택 정렬과 버블 정렬은 제자리 정렬을 할 수 없다.

정답 2번

출제 포인트

대표 정렬 알고리즘의 설계 방식, 시간 복잡도, 안정성, 제자리 정렬 여부를 구별하는 문제이다.

핵심 개념

퀵 정렬과 합병 정렬은 문제를 작은 부분 문제로 나누어 해결한 뒤 결과를 처리하는 분할정복 알고리즘이다.
안정적 정렬은 값이 같은 원소들의 입력 순서를 정렬 후에도 유지하는 정렬이며, 삽입 정렬과 버블 정렬이 대표적이다.
제자리 정렬은 입력 크기에 비례하는 별도의 저장 공간 없이 원래 배열 안에서 정렬하는 방식이며, 선택 정렬과 버블 정렬은 제자리 정렬이 가능하다.

선지별 해설

① X — 선택 정렬의 시간 복잡도는 입력 상태와 관계없이 $O(n^2)$이고, 힙 정렬은 $O(n\log n)$이다. 틀린 표현은 두 알고리즘 모두 $O(n\log n)$이라는 부분이며, 올바르게는 선택 정렬은 $O(n^2)$, 힙 정렬은 $O(n\log n)$이다.
② O — 퀵 정렬은 피벗을 기준으로 부분 배열을 나누어 재귀적으로 정렬하고, 합병 정렬은 배열을 나누어 각각 정렬한 뒤 합병하므로 둘 다 분할정복 방법을 사용한다.
③ X — 삽입 정렬은 일반적으로 안정적이지만, 힙 정렬은 원소의 장거리 교환 과정에서 같은 값의 상대적 순서가 바뀔 수 있어 안정적이지 않다. 틀린 표현은 힙 정렬도 안정적이라는 부분이며, 올바르게는 삽입 정렬은 안정적이고 힙 정렬은 불안정하다.
④ X — 퀵 정렬은 분할이 계속 한쪽으로 치우치면 비교 횟수가 누적되어 최악의 경우 $O(n^2)$이 된다. 틀린 표현은 최악 성능이 $O(n\log n)$이라는 부분이며, 올바르게는 평균적으로 $O(n\log n)$, 최악의 경우 $O(n^2)$이다.
⑤ X — 선택 정렬과 버블 정렬은 모두 원래 배열에서 원소를 교환하므로 제자리 정렬이 가능하다. 틀린 표현은 제자리 정렬을 할 수 없다는 부분이며, 올바르게는 두 정렬 모두 제자리 정렬을 할 수 있다.

오답 포인트

②의 '분할정복'과 ④의 '퀵 정렬 평균 시간 복잡도'를 혼동하기 쉽다. 퀵 정렬은 평균적으로 $O(n\log n)$이지만 분할이 불균형하면 최악의 경우 $O(n^2)$이다. 또한 안정성과 제자리성은 서로 다른 기준이므로 따로 판단해야 한다.

NoteBase

다시 나오면 이렇게 판단

각 선지를 설계 방식, 시간 복잡도, 안정성, 제자리성 중 어느 기준을 묻는지 먼저 분류한다. 퀵·합병은 분할정복, 삽입·버블은 안정적, 선택·버블은 제자리 가능, 퀵의 최악은 $O(n^2)$으로 판단한다.

암기 포인트

퀵·합병은 분할정복, 퀵의 최악은 제곱, 힙은 불안정, 선택·버블은 제자리 가능이다.

NoteBase

16. 다음 글의 ( )안에 공통적으로 들어갈 단어로 옳은 것은?

> 연산이나 부프로그램이 특정 하나의 자료형에 대해 정의된 것이 아니고, 여러 자료형을 처리할 수 있도록 만들어진 경우의 특성을 ( )(이)라고 한다. 이때 특수 범주형 ( )과(와) 매개변수화 ( )(으)로 나눌 수 있다.

① 추상화
② 다형성
③ 캡슐화
④ 상속성
⑤ 정보 은닉

정답 2번

출제 포인트

하나의 연산이나 부프로그램이 여러 자료형을 처리할 수 있는 객체지향 언어의 특성을 구별하는 문제이다.

핵심 개념

다형성은 같은 연산이나 부프로그램이 여러 자료형의 값에 대해 각 자료형에 맞게 동작할 수 있는 성질이다.
제시문에서는 이를 특수 범주형 다형성과 매개 변수화 다형성으로 나눈다.
추상화는 대상의 핵심 속성만 뽑아 표현하는 것이고, 캡슐화는 데이터와 그 데이터를 처리하는 연산을 하나로 묶는 것이다. 상속성은 기존 클래스의 속성과 연산을 다른 클래스가 물려받는 성질이며, 정보 은닉은 객체의 내부 구현을 감추고 허용된 인터페이스를 통해서만 접근하게 하는 원리이다.

선지별 해설

① X — '추상화'가 틀린 표현이다. 추상화는 대상의 불필요한 세부 사항을 제외하고 핵심 속성만 표현하는 것이며, 제시문의 올바른 개념은 '다형성'이다.
② O — 다형성은 하나의 연산이나 부프로그램이 특정 자료형에만 한정되지 않고 여러 자료형을 처리할 수 있는 성질이므로 제시문의 정의와 일치한다.
③ X — '캡슐화'가 틀린 표현이다. 캡슐화는 관련 데이터와 연산을 하나의 단위로 묶는 것이며, 제시문의 올바른 개념은 '다형성'이다.
④ X — '상속성'이 틀린 표현이다. 상속성은 하위 클래스가 상위 클래스의 속성과 연산을 물려받아 재사용하거나 확장하는 것이며, 제시문의 올바른 개념은 '다형성'이다.
⑤ X — '정보 은닉'이 틀린 표현이다. 정보 은닉은 객체의 내부 데이터나 구현을 외부에서 직접 알거나 변경하지 못하도록 접근을 제한하는 것이며, 제시문의 올바른 개념은 '다형성'이다.

NoteBase

오답 포인트

객체지향의 주요 특성들이 함께 제시되어 혼동하기 쉽다. 이 문항의 결정적 표현은 ‘특정 하나의 자료형이 아니고 여러 자료형을 처리한다’는 부분이며, 이는 묶기·물려받기·감추기가 아니라 다형성을 가리킨다.

다시 나오면 이렇게 판단

같은 이름의 연산이나 부프로그램이 자료형에 따라 여러 형태로 동작하면 다형성, 데이터와 연산을 묶으면 캡슐화, 기존 클래스의 구성 요소를 물려받으면 상속성, 내부 구현의 직접 접근을 막으면 정보 은닉으로 판단한다.

암기 포인트

여러 자료형에 하나의 연산이 대응하면 다형성이다.

NoteBase

17. 다음은 정수를 저장할 수 있는 스택을 Java로 구현한 것이다. ㉠과 ㉡에 넣을 문장으로 옳은 것은?

```
public class StackInt {
    int size, top;
    int buf[];
    public StackInt(int s) {
        buf = new int[s];
        size = s;
        top = -1;
    }
    public void push(int x) {
        _____㉠_____;
    }
    public int pop() {
        _____㉡_____;
    }
}
```

	㉠	㉡
①	buf[++top] = x	return buf[--top]
②	buf[top] = x	return buf[top]
③	buf[--top] = x	return buf[top++]
④	buf[++top] = x	return buf[top--]
⑤	buf[top + 1] = x	return buf[top - 1]

정답 4번

출제 포인트

배열로 구현한 스택에서 top의 의미와 push·pop 연산에 필요한 증가·감소 시점을 묻는 문제이다.

핵심 개념

스택은 마지막에 저장한 값을 가장 먼저 꺼내는 LIFO 구조이다. 이 코드에서 top은 현재 스택의 맨 위 원소가 저장된 배열 인덱스이며, 빈 스택에서는 -1이다. 따라서 push는 top을 먼저 1 증가시킨 위치에 저장하는 buf[++top] = x이고, pop은 현재 top 위치의 값을 반환한 뒤 top을 1 감소시키는 return buf[top--]이다.

NoteBase

선지별 해설

① X
㉠의 buf[++top] = x는 맞지만, ㉡의 return buf[--top]은 top을 먼저 감소시켜 현재 맨 위 원소를 건너뛴다. 올바르게는 return buf[top--]이다.

② X
㉠의 buf[top] = x는 빈 스택의 초기 top 값 -1을 그대로 배열 인덱스로 사용하므로 첫 push부터 잘못된다. 또한 ㉡의 return buf[top]은 값을 꺼낸 뒤 top을 감소시키지 않는다. 올바르게는 각각 buf[++top] = x, return buf[top--]이다.

③ X
㉠의 buf[--top] = x는 초기값 -1에서 top을 -2로 감소시켜 잘못된 인덱스를 사용한다. ㉡의 return buf[top++]도 pop 뒤에 top을 증가시키므로 동작 방향이 반대이다. 올바르게는 각각 buf[++top] = x, return buf[top--]이다.

④ O
㉠의 buf[++top] = x는 top을 먼저 증가시킨 뒤 새 원소를 저장하고, ㉡의 return buf[top--]은 현재 맨 위 원소를 반환한 뒤 top을 감소시킨다. 스택의 push와 pop 동작에 맞는다.

⑤ X
㉠의 buf[top + 1] = x는 다음 칸에 값을 쓰지만 top 자체를 증가시키지 않는다. ㉡의 return buf[top - 1]도 현재 맨 위가 아닌 아래쪽 인덱스를 참조하며 top을 갱신하지 않는다. 올바르게는 각각 buf[++top] = x, return buf[top--]이다.

오답 포인트

①은 push 식이 맞아 보이지만 pop에서 --top을 사용한 것이 함정이다. pop은 현재 top의 원소를 먼저 읽어야 하므로 전위 감소가 아니라 후위 감소를 사용한다.

다시 나오면 이렇게 판단

top이 맨 위 원소의 인덱스이고 빈 상태가 -1이면 push는 '증가 후 저장', pop은 '반환 후 감소'로 판단한다. 즉 push는 ++top, pop은 top--의 순서가 된다.

암기 포인트

빈 스택 top = -1: push는 buf[++top], pop은 buf[top--]이다.

NoteBase

18. OSI 7계층 중 두 시스템 사이에서 교환되는 정보의 구문 및 의미와 관련되며, 데이터의 압축 및 암호화를 담당하는 계층으로 옳은 것은?

① 물리 계층(physical layer)
② 데이터링크 계층(data link layer)
③ 네트워크 계층(network layer)
④ 전송 계층(transport layer)
⑤ 표현 계층(presentation layer)

정답 5번

출제 포인트

OSI 7계층에서 데이터의 표현 형식을 변환하고 압축·암호화를 담당하는 계층을 구별하는 문제이다.

핵심 개념

표현 계층은 서로 다른 시스템이 데이터를 같은 의미로 해석할 수 있도록 데이터의 형식과 표현 방식을 변환하는 계층이다. 주요 기능은 코드 변환, 데이터 압축·복원, 암호화·복호화이다.
물리 계층은 비트 전송, 데이터링크 계층은 프레임의 인접 노드 간 전송, 네트워크 계층은 패킷의 경로 선택, 전송 계층은 종단 간 데이터 전송, 표현 계층은 데이터 표현 변환을 담당한다.

선지별 해설

① X — 물리 계층은 전송 매체를 통해 비트를 전기적·광학적 신호로 보내는 계층이다. '데이터의 압축 및 암호화'가 아니라 '비트의 신호 전송'을 담당한다고 고쳐야 한다.
② X — 데이터링크 계층은 프레임 단위의 인접 노드 간 전송, 물리 주소 지정, 오류 검출 등을 담당한다. '데이터의 압축 및 암호화'가 아니라 '프레임의 인접 노드 간 전송'을 담당한다고 고쳐야 한다.
③ X — 네트워크 계층은 논리 주소와 라우팅을 이용해 패킷의 전달 경로를 선택한다. '데이터의 압축 및 암호화'가 아니라 '패킷의 경로 선택과 전달'을 담당한다고 고쳐야 한다.
④ X — 전송 계층은 포트 번호를 이용한 프로세스 간 종단 전송과 신뢰성·흐름 제어 등을 담당한다. '데이터의 압축 및 암호화'가 아니라 '종단 간 데이터 전송'을 담당한다고 고쳐야 한다.
⑤ O — 표현 계층은 데이터의 구문과 의미에 관련된 표현 형식을 맞추며 압축·복원과 암호화·복호화를 담당한다.

오답 포인트

④ 전송 계층의 '전송'이라는 이름 때문에 데이터 처리 기능까지 맡는다고 오해하기 쉽다. 전송 계층은 종단 간 전달을 담당하고, 데이터가 어떤 형식으로 표현되는지와 압축·암호화는 ⑤ 표현 계층이 담당한다.

NoteBase

다시 나오면 이렇게 판단

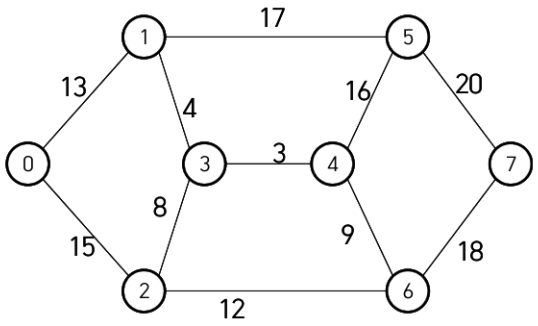
문제에 형식 변환, 문자 코드 변환, 압축·복원, 암호화·복호화가 제시되면 표현 계층으로 판단한다. 경로 선택은 네트워크 계층, 종단 간 전달은 전송 계층으로 구별한다.

암기 포인트

표현 계층은 표현을 맞춘다: 변환·압축·암호화이다.

NoteBase

19. 다음 그래프에서 각 간선에 가중치를 표시하였다. 이 가중치를 근거로 구성한 최소 비용 신장 트리의 총 가중치 합은? (단, 최소 비용 신장 트리 구성은 Kruskal 알고리즘 원리를 적용한다.)



① 67
② 69
③ 71
④ 73
⑤ 75

정답 3번

출제 포인트

Kruskal 알고리즘에 따라 가중치가 작은 간선부터 검사하고, 사이클을 만들지 않는 간선만 선택하여 최소 비용 신장 트리의 총 가중치를 구하는 문제이다.

조건 정리

주어진 값: 간선의 가중치는 3, 4, 8, 9, 12, 13, 15, 16, 17, 18, 20이다.
구해야 할 값: 정점 8개를 모두 연결하는 최소 비용 신장 트리의 총 가중치이다.
주의할 조건: 정점이 8개이므로 간선은 정확히 7개를 선택하며, 선택 과정에서 사이클이 생기는 간선은 제외한다.

사용할 원리

Kruskal 알고리즘은 모든 간선을 가중치의 오름차순으로 살펴보면서, 서로 다른 연결 요소를 잇는 간선만 선택한다. 최소 비용 신장 트리는 모든 정점을 연결하고 사이클이 없는 트리이므로 이 원리를 사용한다.

NoteBase

계산 과정

가중치 3의 간선 (3, 4), 4의 간선 (1, 3), 8의 간선 (2, 3), 9의 간선 (4, 6)을 차례로 선택한다.
가중치 12의 간선 (2, 6)은 이미 연결된 정점 사이에 사이클을 만들므로 제외한다.
가중치 13의 간선 (0, 1)을 선택하고, 가중치 15의 간선 (0, 2)은 사이클을 만들므로 제외한다.
가중치 16의 간선 (4, 5)을 선택하고, 가중치 17의 간선 (1, 5)은 사이클을 만들므로 제외한다.
마지막으로 가중치 18의 간선 (6, 7)을 선택하면 8개 정점이 7개 간선으로 모두 연결된다.
따라서 총 가중치는 3 + 4 + 8 + 9 + 13 + 16 + 18 = 71이다.

선지 판단

① X — '67'은 잘못된 합이다. 사이클을 만들지 않는 7개 간선의 가중치를 모두 더하면 71이므로 올바르게는 '71'이다.
② X — '69'는 잘못된 합이다. 가중치 13, 16, 18의 연결 간선까지 포함한 선택 결과는 71이므로 올바르게는 '71'이다.
③ O — 선택한 간선의 가중치 3, 4, 8, 9, 13, 16, 18의 합이 71이므로 옳다.
④ X — '73'은 잘못된 합이다. 가중치 20의 간선 (5, 7)보다 작은 가중치 18의 간선 (6, 7)로 정점 7을 연결해야 하므로 올바르게는 '71'이다.
⑤ X — '75'는 잘못된 합이다. 가중치가 더 큰 간선을 선택하거나 사이클 간선을 포함한 값이며, Kruskal 알고리즘으로 구한 올바른 총 가중치는 '71'이다.

오답 포인트

가중치가 작은 간선을 무조건 모두 더하면 안 된다. 가중치 12, 15, 17의 간선은 선택 시 사이클을 만들므로 제외해야 한다. 또한 정점 7은 가중치 20이 아니라 더 작은 가중치 18의 간선으로 연결해야 한다.

다시 나오면 이렇게 판단

간선을 가중치 오름차순으로 정렬한 뒤, 사이클 여부를 확인하면서 서로 다른 연결 요소를 잇는 간선만 선택한다. 정점 수가 V이면 간선 $V-1$개가 선택되는 순간 멈추고 그 가중치를 합산한다.

암기 포인트

Kruskal은 '가중치 오름차순 → 사이클 제외 → $V-1$개 선택'이다.

NoteBase

20. 소프트웨어 개발 생명주기 모형 중 나선형 모형에 대한 설명으로 옳지 않은 것은?

① Boehm이 제안한 것으로, 폭포수 모형과 프로토타입 모형의 장점에 위험 분석 기능이 추가된 모형이다.
② 나선을 따라 돌듯이 여러 번의 소프트웨어 개발 과정을 거쳐 점진적으로 최종 소프트웨어를 개발하는 것이다.
③ 프로토타입을 점진적으로 발전시켜, 누락되거나 추가된 요구사항을 반영할 수 있다.
④ 가장 현실적이며, 소규모 시스템에 적합하다.
⑤ 발생할 수 있는 위험을 관리하고 최소화하는 것이 목적이며, 위험성 평가에 크게 의존한다.

정답 4번

출제 포인트

나선형 모형의 반복적 개발 방식, 위험 분석 중심의 특징, 적용 대상과 한계를 구별하는 문제이다.

핵심 개념

나선형 모형: 계획 수립, 위험 분석, 개발 및 검증, 고객 평가의 과정을 나선처럼 반복하면서 소프트웨어를 점진적으로 완성하는 생명주기 모형이다. 위험을 조기에 발견하고 줄이는 데 중점을 두며, 대규모·복잡·고위험 시스템에 적합하다.
폭포수 모형은 단계를 순차적으로 진행하는 방식이고, 프로토타입 모형은 시제품으로 요구사항을 확인하는 방식이다. 나선형 모형은 이들의 장점을 활용하면서 반복 주기마다 위험 분석을 수행한다.

선지별 해설

① O — 나선형 모형은 Boehm이 제안했으며, 폭포수 모형의 체계성과 프로토타입 모형의 반복적 요구 확인에 위험 분석을 결합한 모형이다.
② O — 나선을 따라 여러 번의 개발 주기를 반복하고, 각 주기의 결과를 누적하여 최종 소프트웨어를 점진적으로 완성한다.
③ O — 반복 과정에서 프로토타입을 점진적으로 발전시키므로 초기 단계에서 누락되었거나 이후 추가된 요구사항을 반영할 수 있다.
④ X — 틀린 표현은 '소규모 시스템에 적합하다'이다. 나선형 모형은 위험 분석과 반복 개발에 많은 비용과 관리가 필요하므로 대규모·복잡·고위험 시스템에 적합하며, 소규모 시스템에는 일반적으로 비효율적이다.
⑤ O — 각 반복 주기에서 위험을 분석하고 대응하므로 위험의 관리와 최소화가 핵심 목적이며, 위험을 제대로 식별·평가하는 능력에 크게 의존한다.

NoteBase

오답 포인트

‘가장 현실적’이라는 표현만 보고 ④를 옳다고 판단하기 쉽다. 나선형 모형은 반복과 위험 관리 측면에서 현실적인 장점이 있지만, 위험 분석 비용과 관리 부담이 커서 ‘소규모 시스템에 적합하다’는 부분이 틀리다.

다시 나오면 이렇게 판단

선지에서 ‘반복’, ‘점진적 완성’, ‘프로토타입’, ‘위험 분석’이 함께 나오면 나선형 모형의 특징으로 판단한다. 적용 대상은 소규모가 아니라 대규모·복잡·고위험 시스템이라는 점을 함께 확인한다.

암기 포인트

나선형 모형은 반복 개발과 위험 분석의 결합이며, 대규모·고위험 시스템에 적합하다.

NoteBase