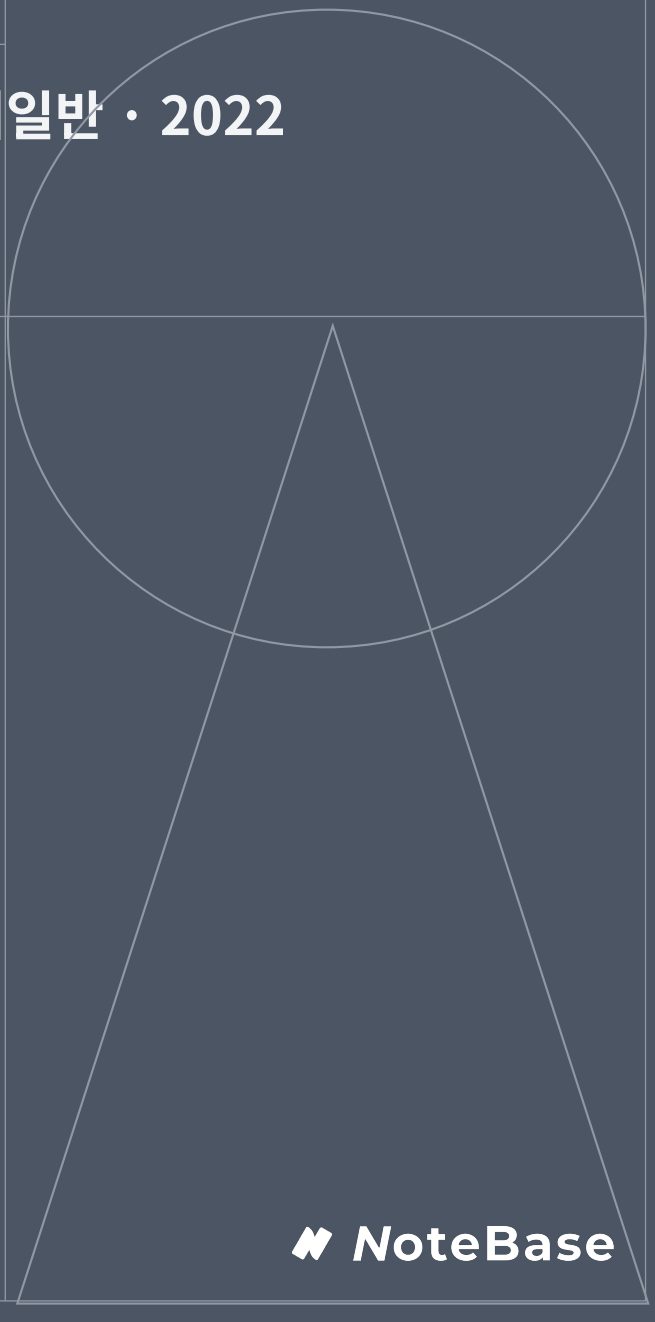
기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2022



NoteBase

기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2022

공무원/지방직/9급 · 컴퓨터일반 · 2022

문 1. 컴퓨터 알고리즘의 조건에 대한 설명으로 옳지 않은 것은?

① 각 명령어의 의미는 모호하지 않고 명확해야 한다.

② 알고리즘 단계들에는 순서가 정해져 있지 않다.

③ 한정된 수의 단계 후에는 반드시 종료되어야 한다.

④ 각 명령어들은 실행 가능한 연산이어야 한다.

정답 2번

간단한 해설

알고리즘은 문제를 해결하기 위한 절차이므로 각 단계를 **어떤 순서와 조건에 따라 실행할지** 정해져 있어야 한다. '단계들에는 순서가 정해져 있지 않다'는 설명은 이와 어긋난다.

선지별 해설

① O: 명확성에 관한 설명이다. 각 명령어는 **의미가 모호하지 않아야** 하며, 무엇을 수행해야 하는지 분명하게 해석할 수 있어야 한다.

② X: 알고리즘은 단계의 **실행 순서와 조건이 명확하게 정해진** 절차이다. 조건에 따라 다른 단계로 이동하거나 같은 단계를 반복할 수 있지만, 그때도 다음에 무엇을 실행할지 결정하는 규칙이 있어야 한다. 따라서 '순서가 정해져 있지 않다'는 '실행 순서가 정해져 있다'로 고쳐야 한다.

③ O: 유한성에 관한 설명이다. 알고리즘은 주어진 입력에 대해 **유한한 수의 단계를 수행한 뒤 종료**되어야 한다. 입력에 따라 필요한 단계 수는 달라질 수 있지만, 끝없이 반복하여 종료하지 않는 절차는 이 조건을 충족하지 못한다.

④ O: 효과성에 관한 설명이다. 각 명령어는 실제로 수행할 수 있는 구체적인 연산이어야 하며, **각 연산을 유한한 시간 안에 수행**할 수 있어야 한다. 이는 전체 절차가 종료되어야 한다는 유한성과 구별되는, 개별 연산의 실행 가능성에 관한 조건이다.

NoteBase

문 2. 다음에서 설명하는 빅데이터의 3대 특징으로 옳지 않은 것은?

빅데이터는 대용량의 데이터 집합으로부터 가치 있는 정보를 효율적으로 추출하고 결과를 분석하는 기술이다.

① 센싱 기술 등을 활용하여 사물과 주위 환경으로부터 정보 획득(sensor)
② 방대한 양의 데이터 처리(volume)
③ 정형 데이터와 비정형 데이터 등 다양한 유형의 데이터로 구성(variety)
④ 실시간으로 생산되며 빠른 속도로 수집 및 분석(velocity)

정답 1번

간단한 해설

센싱(sensor)은 사물과 주변 환경에서 **정보를 수집하는 기술**이다. 이는 데이터의 규모·유형·발생 속도를 나타내는 빅데이터의 3대 특징에 포함되지 않는다.

선지별 해설

① X: 센싱(sensor)은 센서 등을 이용하여 사물과 주변 환경의 정보를 얻는 기술이다. 빅데이터를 분석하기 위한 데이터 수집에 활용될 수 있지만, **빅데이터의 3대 특징에는 포함되지 않는다**. 빅데이터의 3대 특징은 규모(volume), 다양성(variety), 속도(velocity)이다.

② O: 규모(volume)는 **데이터의 양이 방대하다**는 특징이다. 대용량 데이터를 저장하고 처리해야 한다는 점을 나타내므로, 방대한 양의 데이터 처리라는 설명에 대응한다.

③ O: 다양성(variety)은 **데이터의 유형과 형태가 다양하다**는 특징이다. 일정한 형식의 표에 저장되는 정형 데이터뿐 아니라 문서·이미지·영상과 같은 비정형 데이터도 분석 대상이 된다.

④ O: 속도(velocity)는 **데이터가 빠르게 생성·유입된다**는 특징이다. 계속 들어오는 데이터에서 필요한 정보를 제때 얻으려면 빠른 수집과 처리가 필요하며, 실시간 분석도 이 특징과 관련된다.

NoteBase

문 3. 다음 자료를 오름차순으로 삽입 정렬(insertion sort)하는 과정에서 나올 수 없는 경우는?

3 1 4 2 9 5

① 1 3 4 2 9 5
② 1 2 3 4 9 5
③ 3 1 5 2 4 9
④ 1 2 3 4 5 9

정답 3번

간단한 해설

삽입 정렬은 **앞부분을 정렬한 뒤 다음 원소를 삽입**한다. '3 1 5 2 4 9'처럼 맨 앞의 3과 1이 정렬되지 않은 채 뒤쪽의 5가 앞으로 이동한 상태는 나올 수 없다.

공통 해설

오름차순 삽입 정렬은 첫 원소를 정렬된 구간으로 보고, 바로 다음 원소를 그 구간의 알맞은 위치에 넣는 과정을 반복한다. 한 원소의 삽입이 끝날 때마다 **처음부터 처리한 원소까지는 오름차순**이며, 아직 처리하지 않은 뒤쪽 원소들은 원래 순서와 위치를 유지한다.

주어진 자료에서 처음 두 원소 3, 1을 정렬하면 '1 3 4 2 9 5'가 된다. 같은 원리에 따라 처음 네 원소 3, 1, 4, 2까지 처리하면 '1 2 3 4 9 5'가 되고, 마지막 원소 5까지 처리하면 '1 2 3 4 5 9'가 된다. 이들은 모두 앞부분의 정렬 구간을 차례로 넓혀 가는 과정에서 가능한 상태이다.

반면 '3 1 5 2 4 9'에서는 맨 뒤에 있던 5가 이동했는데도 앞의 3과 1이 내림차순으로 남아 있다. 5를 삽입할 때는 그보다 앞의 원소들이 이미 정렬되어 있어야 하므로 이러한 상태는 나올 수 없다.

NoteBase

문 4. 소프트웨어의 화이트박스 테스트에 대한 설명으로 옳지 않은 것은?

① 글래스 박스(Glass-box) 테스트라고 부른다.

② 소프트웨어의 내부 경로에 대한 지식을 보지 않고 테스트 대상의 기능이나 성능을 테스트하는 기술이다.

③ 문장 커버리지, 분기 커버리지, 조건 커버리지 등의 검증 기준이 있다.

④ 모듈의 논리적인 구조를 체계적으로 점검하기 때문에 구조적 테스트라고도 한다.

정답 2번

간단한 해설

②의 '내부 경로에 대한 지식을 보지 않고' 기능이나 성능을 검사한다는 설명은 **블랙박스 테스트**에 해당한다. 화이트박스 테스트는 **내부 코드와 논리 구조**를 바탕으로 실행 경로를 검사한다.

선지별 해설

① O: 화이트박스 테스트는 프로그램의 내부 코드와 처리 구조를 들여다보며 검사한다. 내부가 보이는 유리 상자에 빗대어 **글래스 박스(Glass-box) 테스트**라고도 부른다.

② X: 내부 경로를 고려하지 않고 요구사항에 따라 기능이나 성능을 검사하는 것은 **블랙박스 테스트**이다. 화이트박스 테스트는 내부 코드와 제어 흐름을 분석하여 검사할 경로와 테스트 데이터를 정한다. 예를 들어 조건문이 있으면 입력값을 달리하여 참인 경우와 거짓인 경우의 실행 흐름을 검사한다.

③ O: 커버리지는 테스트가 정해진 검사 대상을 얼마나 실행했는지를 나타내는 기준이다. 문장 커버리지는 실행 가능한 각 문장을 실행했는지, 분기 커버리지는 각 분기의 결과를 실행했는지, 조건 커버리지는 각 개별 조건이 참과 거짓을 모두 갖도록 검사했는지를 확인한다. 이들은 **내부 코드의 실행 범위**를 평가하는 화이트박스 테스트의 검증 기준이다.

④ O: 화이트박스 테스트는 모듈 내부의 조건문·반복문과 같은 **논리 구조와 제어 흐름**을 기준으로 체계적으로 검사한다. 외부에서 관찰되는 입출력만 보는 것이 아니라 내부 구조를 바탕으로 검사하므로 구조적 테스트라고도 한다.

NoteBase

문 5. 16진수 210을 8진수로 변환한 것은?

① 1020

② 2100

③ 10210

④ 20100

정답 1번

간단한 해설

16진수를 2진수로 바꾼 뒤 **오른쪽부터 세 자리씩** 묶으면 001·000·010·000이 된다. 각 묶음을 8진수 한 자리로 바꾸면 1020이다.

계산 과정

1. 16진수를 2진수로 변환한다. $16 = 2^4$이므로 16진수 한 자리는 2진수 네 자리에 대응한다. 각 숫자를 **네 자리를 유지하여** 바꾸면 2는 0010, 1은 0001, 0은 0000이다. 이를 이어 붙이면 $(210)_{16} = (001000010000)_2$이다.

2. 2진수를 세 자리씩 묶는다. $8 = 2^3$이므로 2진수 세 자리가 8진수 한 자리에 대응한다. 정수 부분은 **오른쪽 끝을 기준으로** 묶어야 각 자리의 값이 유지된다. 따라서 위의 2진수를 $(001000010000)_2$으로 다시 묶는다.

3. 각 묶음을 8진수로 변환한다. 001은 1, 000은 0, 010은 2, 마지막 000은 0이므로 차례대로 읽으면 1020이다. 따라서 $(210)_{16} = (1020)_8$이며, 이는 ①의 1020에 해당한다.

NoteBase

문 6. 은행원 알고리즘(banker’s algorithm)이 교착상태를 해결하는 방법은?

① 예방

② 회피

③ 검출

④ 회복

정답 2번

간단한 해설

은행원 알고리즘은 자원을 할당하기 전에 할당 후에도 **안전 상태가 유지되는지** 검사한다. 교착상태의 위험이 있는 할당을 미루므로 교착상태 **회피**에 해당한다.

개념 구분

은행원 알고리즘: 각 프로세스가 요구할 수 있는 최대 자원량을 미리 알고 있다는 전제에서 자원 요청을 심사하는 **교착상태 회피** 알고리즘이다. 요청한 자원을 할당했다고 가정한 뒤, 모든 프로세스가 필요한 자원을 받아 작업을 마치고 자원을 반환할 수 있는 순서가 있는지 검사한다. 이러한 순서가 존재하는 상태를 '안전 상태'라고 한다. 할당 후에도 안전 상태이면 자원을 할당하고, 그렇지 않으면 요청한 프로세스를 대기시켜 불안전 상태로의 진입을 피한다.

예방과의 구분: 교착상태 예방은 상호 배제·점유와 대기·비선점·순환 대기라는 네 필요조건 중 **하나 이상이 성립하지 않도록** 자원 할당 규칙을 정하는 방법이다. 회피는 최대 자원 요구량과 현재 할당 상태를 이용해 각 요청의 안전성을 판단한다는 점에서 예방과 구별된다. 검출은 이미 교착상태가 발생했는지 찾아내는 것이고, 회복은 발생한 교착상태를 프로세스 종료나 자원 선점 등으로 해소하는 것이다.

NoteBase

문 7. 다음 OSI 7계층 중 물리 계층에 해당하는 장치를 모두 고른 것은?

ㄱ. 리피터(Repeater)
ㄴ. 더미허브(Dummy Hub)
ㄷ. 라우터(Router)
ㄹ. 게이트웨이(Gateway)
ㅁ. 브릿지(Bridge)

① ㄱ, ㄴ
② ㄱ, ㄷ
③ ㄴ, ㄹ
④ ㄹ, ㅁ

정답 1번

간단한 해설

리피터와 더미허브는 주소를 보고 전송 대상을 판단하지 않고 **비트 신호를 재생·중계**하므로 물리 계층 장치이다. 브릿지는 MAC 주소를, 라우터는 IP 주소를 이용한다는 점에서 구별된다.

보기별 해설

ㄱ. 리피터: **물리 계층(1계층)** 장치이다. 전송 과정에서 약해지거나 일그러진 신호를 재생하여 더 멀리 전달한다. 프레임이나 패킷의 목적지 주소를 해석해 경로를 선택하지 않는다.

ㄴ. 더미허브: 여러 포트를 가진 리피터로, **물리 계층(1계층)**에서 동작한다. 한 포트로 들어온 신호를 수신 포트를 제외한 다른 모든 포트로 전달하며, 목적지 주소에 따라 특정 포트만 선택하지 않는다.

ㄷ. 라우터: **네트워크 계층(3계층)** 장치이다. 목적지 IP 주소와 라우팅 정보를 이용하여 패킷을 보낼 경로를 결정하므로, 비트 신호를 중계하는 물리 계층 장치와 다르다.

ㄹ. 게이트웨이: 서로 다른 프로토콜을 사용하는 네트워크 사이에서 **프로토콜을 변환**하여 통신을 가능하게 하는 장치이다. 단순히 신호를 재생·중계하는 물리 계층 장치로 분류하지 않는다.

ㅁ. 브릿지: **데이터 링크 계층(2계층)** 장치이다. MAC 주소를 이용하여 프레임을 다른 네트워크 구간으로 전달할지 판단하므로, 주소를 해석하지 않는 리피터나 더미허브와 구별된다.

물리 계층에 해당하는 장치는 ㄱ과 ㄴ이며, 이 조합은 ①에 해당한다.

NoteBase

문 8. 이미지 표현을 위한 RGB 방식과 CMYK 방식에 대한 설명으로 옳은 것은?

① CMYK 방식은 가산 혼합 모델로 빛이 하나도 없을 때 검은색을 표현한다.

② CMYK 방식에서 C는 Cyan을 의미한다.

③ RGB 방식은 주로 컬러 프린터, 인쇄, 페인팅 등에 적용된다.

④ RGB 방식에서 B는 Black을 의미한다.

정답 2번

간단한 해설

CMYK는 청록·자홍·노랑·검정 잉크로 색을 표현하는 방식이며, **C는 Cyan(청록색)**을 뜻한다.

선지별 해설

① X: CMYK는 잉크가 빛의 일부를 흡수하여 색을 표현하는 **감산 혼합 모델**이다. 흰 종이에 잉크를 칠하지 않으면 종이의 흰색이 보인다. 빛을 더해 색을 표현하고 빛이 하나도 없을 때 검은색이 되는 것은 RGB의 가산 혼합 모델이다.

② O: CMYK의 **C는 Cyan(청록색)**을 뜻한다. M은 Magenta(자홍색), Y는 Yellow(노란색), K는 검은색 잉크를 나타낸다.

③ X: RGB는 빨강·초록·파랑 빛을 더해 색을 표현하므로 주로 **모니터 등 화면 표시**에 사용된다. 컬러 프린터나 인쇄처럼 잉크를 사용하는 색 표현에는 주로 CMYK 방식이 적용된다. 물감을 섞는 페인팅도 빛의 가산 혼합이 아니라 감산 혼합에 해당한다.

④ X: RGB에서 **B는 Blue(파란색)**를 뜻한다. R은 Red(빨간색), G는 Green(초록색)이며, 검은색은 세 색의 빛이 모두 없는 상태로 표현한다. CMYK에서 검은색 잉크를 나타내는 문자는 K이다.

NoteBase

문 9. 다음은 A 계좌에서 B 계좌로 3,500원을 이체하는 계좌 이체 트랜잭션 T_1과, C 계좌에서 D 계좌로 5,200원을 이체하는 계좌 이체 트랜잭션 T_2가 순차적으로 수행되면서 기록된 로그파일 내용이다. (가)의 시점에서 장애가 발생했을 경우 지연 갱신 회복 기법을 적용했을 때 트랜잭션에 대한 회복조치로 옳은 것은?

1: <T_1, start>
2: <T_1, A, 7800>
3: <T_1, B, 3500>
4: <T_1, commit>
5: <T_2, start>
6: <T_2, C, 9820>
_____(가)_____
7: <T_2, D, 5200>
8: <T_2, commit>

① T_1, T_2 트랜잭션 모두 별다른 조치를 수행하지 않는다.
② T_1 트랜잭션의 로그 내용을 무시하고 버린다.
③ T_1 트랜잭션에는 별다른 회복조치를 하지 않지만, T_2 트랜잭션에는 redo(T_2) 연산을 실행한다.
④ T_2 트랜잭션에는 별다른 회복조치를 하지 않지만, T_1 트랜잭션에는 redo(T_1) 연산을 실행한다.

정답 4번

간단한 해설

지연 갱신은 **커밋 전에는 변경 내용을 데이터베이스에 반영하지 않는** 방식이다. 장애 시점에 커밋된 T_1은 로그로 재실행하고, 아직 커밋되지 않은 T_2는 데이터베이스에 반영된 변경이 없어 취소 연산이 필요 없다.

NoteBase

공통 해설

지연 갱신 회복 기법은 트랜잭션 실행 중의 변경 내용을 로그에 기록하되, 데이터베이스에 반영하는 것은 커밋 이후로 미룬다. 따라서 장애 복구 시에는 **커밋 기록이 있는 트랜잭션의 변경 내용을 redo로 재반영**하고, 커밋 기록이 없는 트랜잭션의 변경 내용은 반영하지 않는다. 미완료 트랜잭션의 변경은 데이터베이스에 기록되지 않았으므로 이를 되돌리는 undo도 필요 없다.

(가)는 6번 로그 다음이므로, T_1의 4번 commit 기록은 있지만 T_2의 8번 commit 기록은 없다. T_1은 완료된 이체 결과를 보존해야 하므로 redo(T_1)을 수행하여 로그에 기록된 A 계좌의 7,800원과 B 계좌의 3,500원을 데이터베이스에 반영한다. 커밋 기록이 있다는 사실과 변경 내용이 데이터베이스에 모두 반영되었다는 사실은 구별해야 한다.

T_2는 C 계좌의 변경 내용을 로그에 기록했지만 커밋 전에 장애가 발생했다. 지연 갱신에서는 이 변경이 데이터베이스에 반영되지 않았으므로 **T_2에는 redo와 undo 모두 필요 없다**. 즉, T_1에는 재실행을 적용하고 T_2에는 별도의 데이터베이스 회복 연산을 수행하지 않는다.

NoteBase

문 10. 다음에 해당하는 CMMI(Capability Maturity Model Integration) 모델의 성숙 단계로 옳은 것은? (단, 하위 성숙 단계는 모두 만족한 것으로 가정한다)

◯ 요구사항 개발
◯ 조직 차원의 프로세스 정립
◯ 기술적 솔루션
◯ 조직 차원의 교육훈련
◯ 제품 통합
◯ 통합 프로젝트 관리
◯ 검증
◯ 위험관리
◯ 확인
◯ 의사 결정 분석 및 해결
◯ 조직 차원의 프로세스 개선

① 2단계
② 3단계
③ 4단계
④ 5단계

정답 2번

간단한 해설

조직 전체의 표준 프로세스를 정립하고 교육훈련과 통합 프로젝트 관리에 적용하는 것은 **3단계인 정의 단계(Defined)**의 특징이다. 지문의 프로세스 영역들은 개별 프로젝트 관리를 넘어 조직 차원의 일관된 개발 절차를 갖추는 데 초점을 둔다.

NoteBase

핵심 풀이

3단계인 정의 단계(Defined)는 **조직 전체의 표준 프로세스**를 수립하고, 각 프로젝트가 이를 프로젝트 특성에 맞게 조정하여 사용하는 단계이다. 지문의 '조직 차원의 프로세스 정립', '조직 차원의 교육훈련', '통합 프로젝트 관리'가 이러한 조직 차원의 표준화와 적용을 보여 준다.

요구사항 개발, 기술적 솔루션, 제품 통합, 검증, 확인은 3단계의 개발 관련 프로세스 영역이다. 위험관리와 의사결정 분석 및 해결도 3단계에 속한다. '조직 차원의 프로세스 개선' 역시 조직 프로세스의 강점과 약점을 파악하고 개선 활동을 계획·실행하는 3단계 영역으로, **'개선'이라는 단어만으로 5단계를 뜻하지는 않는다**.

2단계인 관리 단계는 개별 프로젝트의 계획·수행·통제에 필요한 관리 체계를 갖추는 데 초점을 둔다. 이에 비해 3단계에서는 조직 공통의 표준 프로세스를 바탕으로 프로젝트 간 일관성을 확보한다. 4단계인 정량적 관리 단계는 **통계적·정량적 기법으로 품질과 프로세스 성과를 관리**하며, 5단계인 최적화 단계는 정량적 이해를 바탕으로 프로세스 성과를 지속적으로 개선한다.

문제에서 하위 성숙 단계를 모두 만족한다고 가정했으므로, 제시된 프로세스 영역이 속한 수준을 판단하면 된다. 제시된 영역의 조합은 3단계인 정의 단계에 해당한다.

NoteBase

문 11. 다음은 정논리를 사용하는 JK 플립플롭의 진리표이다. (가) ~ (라)에 들어갈 내용으로 옳은 것은? (단, Q'은 Q의 반댓값을 의미한다)

CP	J	K	다음상태 Q
↑	0	0	(가)
↑	0	1	(나)
↑	1	0	(다)
↑	1	1	(라)

① (가): Q, (나): 1, (다): 0, (라): Q′

② (가): Q′, (나): 1, (다): 0, (라): Q

③ (가): Q, (나): 0, (다): 1, (라): Q′

④ (가): Q′, (나): 0, (다): 1, (라): Q

정답 3번

간단한 해설

JK 플립플롭에서 J와 K가 모두 0이면 **현재 상태를 유지**하고, 모두 1이면 **현재 상태를 반전**한다. J만 1이면 다음 상태가 1로, K만 1이면 0으로 바뀐다.

NoteBase

빈칸별 해설

표의 CP 열에 있는 ↑는 클록이 0에서 1로 바뀌는 상승 에지를 뜻한다. 이때 J와 K의 조합에 따라 다음 상태가 결정되며, Q는 현재 상태, Q′은 현재 상태의 반댓값을 나타낸다.

(가) Q: J = 0, K = 0이면 **현재 상태를 유지**한다. 현재 Q가 0이면 다음 상태도 0이고, 현재 Q가 1이면 다음 상태도 1이므로 Q가 들어간다.

(나) 0: J = 0, K = 1이면 **리셋** 동작을 한다. 현재 Q가 0인지 1인지와 관계없이 다음 상태를 0으로 만든다.

(다) 1: J = 1, K = 0이면 **세트** 동작을 한다. 현재 Q와 관계없이 다음 상태를 1로 만든다.

(라) Q′: J = 1, K = 1이면 **토글** 동작으로 현재 상태를 반전한다. 현재 Q가 0이면 다음 상태는 1이고, 현재 Q가 1이면 다음 상태는 0이므로 Q′이 들어간다.

따라서 (가) Q, (나) 0, (다) 1, (라) Q′의 조합이므로 ③에 해당한다.

NoteBase

문 12. 다음 SQL(Structured Query Language)문으로 생성한 테이블에 내용을 삽입할 때 올바르게 동작하지 않는 SQL 문장은?

```
CREATE TABLE Book (ISBN CHAR(17) PRIMARY KEY,
TITLE VARCHAR(30) NOT NULL, PRICE INT NOT NULL,
PUBDATE DATE, AUTHOR VARCHAR(30));
```

① INSERT INTO Book (ISBN, TITLE, PRICE, AUTHOR) VALUES ('978-89-8914-892-1', '데이터베이스 개론', 20000, '홍길동');

② INSERT INTO Book VALUES ('978-89-8914-892-2', '데이터베이스 개론', 20000, '2022-06-18', '홍길동');

③ INSERT INTO Book (ISBN, TITLE, PRICE) VALUES ('978-89-8914-892-3', '데이터베이스 개론', 20000);

④ INSERT INTO Book (ISBN, TITLE, AUTHOR) VALUES ('978-89-8914-892-4', '데이터베이스 개론', '홍길동');

정답 4번

간단한 해설

PRICE는 **NOT NULL이며 기본값도 없으므로 생략할 수 없다**. ISBN, TITLE, AUTHOR만 지정하면 PRICE에 넣을 값이 없어 제약조건을 위반한다.

NoteBase

공통 해설

INSERT 문에서 생략한 열에는 지정된 기본값이 들어가며, 기본값이 없으면 NULL이 들어간다. 따라서 **NULL을 허용하지 않고 기본값도 없는 열은 반드시 값을 제공**해야 한다. 제시된 Book 테이블에서는 기본키인 ISBN과 NOT NULL로 선언된 TITLE, PRICE가 이에 해당한다. PUBDATE와 AUTHOR는 NOT NULL 제약조건이 없어 생략할 수 있다.

모든 후보는 ISBN과 TITLE을 제공하므로, 이 문항에서는 PRICE의 제공 여부가 삽입 가능 여부를 가른다. 열 목록에 ISBN, TITLE, PRICE, AUTHOR를 지정하면 PUBDATE만 생략되고, ISBN, TITLE, PRICE를 지정하면 PUBDATE와 AUTHOR가 생략된다. 두 경우 모두 필수 열의 값은 제공된다. 열 목록 없이 다섯 값을 제공한 문장도 테이블의 선언 순서인 ISBN, TITLE, PRICE, PUBDATE, AUTHOR에 각각 대응하므로 필수 열을 빠뜨리지 않는다.

반면 ISBN, TITLE, AUTHOR만 지정한 문장은 PRICE를 생략한다. PRICE에는 기본값이 없으므로 NULL이 들어가려 하지만 NOT NULL 제약조건 때문에 삽입이 실패한다. 이 문장을 실행하려면 열 목록에 PRICE를 추가하고, VALUES의 대응 위치에도 20000과 같은 가격 값을 추가해야 한다.

NoteBase

문 13. 패킷 교환 네트워크에 대한 설명으로 옳지 않은 것은?

① 패킷 크기는 옥텟(Octet) 단위로 사용한다.

② 네트워크로 전송되는 모든 데이터는 송·수신지 정보를 포함하는 패킷들로 구성된다.

③ 패킷 교환 방식은 접속 방식에 따라 데이터그램 방식과 가상회선 방식이 있다.

④ 패킷 교환 네트워크에서는 동시에 2쌍 이상의 통신이 불가능하다.

정답 4번

간단한 해설

④의 '동시에 2쌍 이상의 통신이 불가능하다'는 설명은 잘못되었다. 패킷 교환에서는 여러 통신의 패킷이 전송 회선을 나누어 사용하므로 **여러 쌍이 동시에 통신할 수 있다**.

선지별 해설

① O: 옥텟(Octet)은 **8비트**를 뜻하며, 패킷의 길이나 크기를 나타내는 단위로 사용한다. 예를 들어 100옥텟은 800비트이다.

② O: 패킷 교환 네트워크에서는 전송 데이터를 패킷 단위로 나누어 보낸다. 패킷에는 실제 데이터뿐 아니라 **송·수신지를 식별하고 전달하는 데 필요한 제어 정보**가 포함된다. 이 정보는 방식에 따라 주소나 연결 식별 정보 등으로 표현된다.

③ O: 패킷 교환은 **데이터그램 방식과 가상회선 방식**으로 구분한다. 데이터그램 방식은 사전에 연결을 설정하지 않고 각 패킷을 독립적으로 전달하므로 패킷마다 경로가 달라질 수 있다. 가상회선 방식은 먼저 논리적 연결을 설정한 뒤 그 연결을 따라 패킷을 전달한다.

④ X: 패킷 교환에서는 통신하는 한 쌍이 전송 회선을 계속 독점하지 않고, **여러 통신의 패킷이 회선을 공유**한다. 예를 들어 한 회선에서 A와 B의 패킷 사이에 C와 D의 패킷을 끼워 전송하면 두 쌍의 통신이 함께 진행된다. 따라서 '동시에 2쌍 이상의 통신이 불가능하다'는 '동시에 2쌍 이상의 통신이 가능하다'로 고친다.

NoteBase

문 15. 다음 C 프로그램을 실행하면서 사용자가 1, 2, 3, 4를 차례대로 입력했을 때, 출력 결과는?

```
#include <stdio.h>

int main()
{
   int ary[4];
   int sum = 0;
   int i;

   for (i = 0; i < 4; i++) {
      printf("%d번 째 값을 입력하시오 : ", i + 1);
      scanf("%d", &ary[i]);
   }

   for (i = 3; i > 0; i--)
      sum += ary[i];

   printf("%d \n", sum);
   return 0;
}
```

① 3

② 6

③ 9

④ 10

정답 3번

간단한 해설

합계를 구하는 반복문은 **i > 0**일 때만 실행되므로 ary[0]에 저장된 1은 더하지 않는다. ary[3], ary[2], ary[1]의 값만 더해 출력값은 4 + 3 + 2 = 9가 된다.

NoteBase

계산 과정

입력과 초기값: 첫 번째 반복문은 i가 0, 1, 2, 3일 때 차례로 입력을 받는다. 따라서 ary[0] = 1, ary[1] = 2, ary[2] = 3, ary[3] = 4가 저장되며, 합계를 담는 sum의 초기값은 0이다.

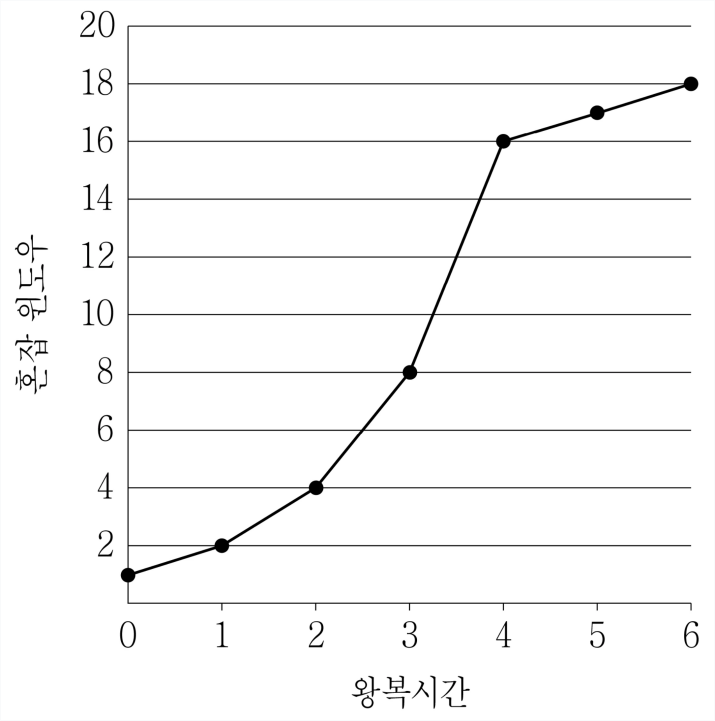
합산 범위: 두 번째 반복문은 i = 3에서 시작하여 매번 i를 1씩 줄인다. 조건이 **i > 0**이므로 i가 3, 2, 1일 때만 sum += ary[i]를 실행한다. i가 0이 되면 반복을 끝내므로 ary[0]은 합산에서 제외된다.

누적 계산: sum += ary[i]는 현재 sum에 ary[i]의 값을 더해 다시 sum에 저장한다는 뜻이다. i = 3일 때 sum은 0 + 4 = 4, i = 2일 때 4 + 3 = 7, i = 1일 때 7 + 2 = 9가 된다.

출력: 마지막 printf는 sum의 최종값인 9를 출력하고 줄을 바꾼다. 따라서 출력값에 해당하는 선택지는 ③의 9이다.

NoteBase

문 16. 그림은 TCP Tahoe에서 데이터 전송에 따른 혼잡 윈도우(cwnd, 단위: MSS)의 크기 변화를 나타낸다. 혼잡 윈도우값이 18일 때의 전송에서 Time-out이 발생했을 때, 느린 출발(slow-start) 임곗값과 혼잡 윈도우값 변화로 옳은 것은?



① 임곗값은 변하지 않고, 혼잡 윈도우값은 1로 감소한다.

② 임곗값이 9가 되고, 혼잡 윈도우값은 1로 감소한다.

③ 임곗값이 9가 되고, 혼잡 윈도우값은 현재 값의 반으로 감소한다.

④ 임곗값은 변하지 않고, 혼잡 윈도우값은 현재 값의 반으로 감소한다.

정답 2번

간단한 해설

TCP Tahoe에서 Time-out이 발생하면 **임곗값은 직전 혼잡 윈도우의 절반**으로 갱신하고, **혼잡 윈도우는 1 MSS**로 줄여 느린 출발을 다시 시작한다. 직전 값이 18 MSS이므로 새 임곗값은 9 MSS이다.

NoteBase

공통 해설

TCP Tahoe는 Time-out을 혼잡의 신호로 보고, 느린 출발 임곗값(ssthresh)을 직전 혼잡 윈도우의 절반으로 갱신하면서 혼잡 윈도우(cwnd)를 1 MSS로 줄인다. 이는 전송량을 크게 줄인 뒤 느린 출발로 다시 증가시키는 혼잡 제어 규칙이다. ssthresh는 느린 출발에서 혼잡 회피로 전환하는 기준이고, cwnd는 확인 응답을 기다리지 않고 보낼 수 있는 데이터양을 제한하는 값이다.

그림의 마지막 점에서 cwnd는 18 MSS이다. 따라서 새 임곗값은 $\frac{18}{2} = 9$ MSS이고, cwnd는 1 MSS로 감소한다. 여기서 **직전 cwnd의 절반은 새 임곗값**이지, 감소 후의 cwnd가 아니다. 기존 임곗값을 유지하는 것도 이 규칙에 맞지 않는다.

NoteBase

문 17. 다중 프로그래밍 환경에서 연속 메모리 할당 방법에 대한 설명으로 옳지 않은 것은?

① 가변분할 메모리 할당은 프로세스의 크기에 따라 메모리를 나누는 것으로 단편화 문제가 발생하지 않는다.

② 가변분할 메모리 할당의 메모리 배치방법으로는 최초 적합, 최적 적합, 최악 적합 방법이 있다.

③ 고정분할 메모리 할당은 프로세스의 크기와 상관없이 메모리를 같은 크기로 나누는 것이다.

④ 고정분할 메모리 할당에서는 쓸모없는 공간으로 인해 메모리 낭비가 발생할 수 있다.

정답 1번

간단한 해설

가변분할은 프로세스 크기에 맞춰 메모리를 할당하지만, 할당과 해제가 반복되면 빈 공간이 여러 곳에 흩어지는 **외부 단편화가 발생할 수 있다**. 따라서 '단편화 문제가 발생하지 않는다'는 설명이 잘못되었다.

선지별 해설

① X: 가변분할은 프로세스가 필요한 크기에 맞춰 연속된 메모리 영역을 할당한다. 프로세스가 종료되어 영역이 해제되면 사용 중인 영역 사이에 작은 빈 공간들이 남을 수 있다. 이처럼 빈 공간의 총량은 충분해도 하나의 연속 영역이 부족해 할당하지 못하는 문제가 **외부 단편화**이다. 따라서 '단편화 문제가 발생하지 않는다'는 '외부 단편화가 발생할 수 있다'로 고쳐야 한다.

② O: 가변분할에서는 프로세스를 넣을 빈 영역을 선택하는 배치 방법이 필요하다. 최초 적합은 탐색 순서상 처음 발견한 충분히 큰 영역을, 최적 적합은 충분히 큰 영역 중 **가장 작은 영역**을, 최악 적합은 가장 큰 영역을 선택한다. 최적 적합과 최악 적합은 각각 할당 직후 남는 공간을 가장 작게 하거나 가장 크게 하는 방식이다.

③ O: 선지는 메모리를 같은 크기로 미리 나누는 **균등 고정분할** 방식을 설명한다. 이 방식에서는 프로세스가 들어올 때마다 그 크기에 맞춰 분할 크기를 바꾸지 않고, 미리 정한 분할 영역에 프로세스를 배치한다. 다만 고정분할에는 서로 다른 크기의 영역을 미리 정하는 비균등 방식도 있으므로, 모든 고정분할의 영역 크기가 반드시 같은 것은 아니다. 고정분할의 핵심은 분할 크기와 경계를 미리 정해 둔다는 점이다.

④ O: 고정분할에서는 프로세스가 자신에게 배정된 분할 영역보다 작으면 **할당된 영역 내부에 사용하지 않는 공간**이 남는다. 예를 들어 100 KB 영역에 70 KB 프로세스를 넣으면 남은 30 KB를 다른 프로세스에 따로 할당하지 못한다. 이러한 낭비를 내부 단편화라고 한다.

NoteBase

문 18. 병렬 프로세서에 대한 설명으로 옳지 않은 것은?

① 프로세스 수준 병렬성은 다수의 프로세서를 이용하여 독립적인 프로그램 여러 개를 동시에 수행한다.

② 클러스터는 근거리 네트워크를 통하여 연결된 컴퓨터들이 하나의 대형 멀티 프로세서로 동작하는 시스템이다.

③ 공유 메모리 프로세서(SMP)는 단일 실제 주소 공간을 갖는 병렬 프로세서를 의미한다.

④ 각 프로세서의 메모리 접근법 분류에 따르면 UMA는 약결합형 다중처리기 시스템, NUMA 및 NORMA는 강결합형 다중처리기 시스템에 해당한다.

정답 4번

간단한 해설

④는 강결합형과 약결합형의 분류를 잘못 연결했다. 메모리 공유 방식에 따른 분류에서 **UMA와 NUMA는 강결합형**이고, **NORMA는 약결합형**이다.

선지별 해설

① O: 프로세스 수준 병렬성은 **서로 독립적인 프로그램이나 프로세스**를 여러 프로세서에 나누어 동시에 실행하는 방식이다. 예를 들어 한 프로세서가 문서 처리 프로그램을 실행하는 동안 다른 프로세서가 별도의 계산 프로그램을 실행할 수 있다.

② O: 클러스터는 독립된 컴퓨터들을 근거리 네트워크로 연결하여 **하나의 시스템처럼 협력하여 작업을 수행**하게 하는 구성이다. 각 컴퓨터가 작업의 일부를 맡아 처리하므로 전체적으로 하나의 대형 다중프로세서 시스템과 같은 역할을 할 수 있다.

③ O: 공유 메모리 방식의 병렬 시스템에서는 여러 프로세서가 **단일 물리 주소 공간**을 공유한다. 따라서 서로 다른 프로세서가 같은 공유 메모리 위치에 접근하여 데이터를 주고받을 수 있다. 메모리가 물리적으로 여러 곳에 나뉘어 있더라도 하나의 공유 주소 공간을 제공할 수 있다.

④ X: UMA와 NUMA는 공유 메모리 기반의 강결합형이고, NORMA는 각 프로세서의 메모리가 분리된 약결합형이다. UMA는 어느 프로세서가 공유 메모리에 접근하더라도 접근 시간이 균일하다. NUMA는 메모리 위치에 따라 접근 시간이 달라지지만, **하나의 공유 주소 공간을 제공하므로 강결합형**에 속한다. NORMA는 다른 프로세서의 메모리를 일반적인 메모리 접근으로 직접 읽고 쓸 수 없어 메시지 전달로 통신하는 **약결합형**이다.

NoteBase

문 19. 다음 C 프로그램의 실행 결과로 옳은 것은?

```
#include <stdio.h>

int star = 10;

void printStar() {
   printf("%d \n", star);
}

int main()
{
   int star = 5;

   printStar();
   printf("%d \n", star);
   return 0;
}
```

① 5
5

② 5
10

③ 10
5

④ 10
10

정답 3번

간단한 해설

main의 지역 변수 star는 **main 안에서만 전역 변수를 가린다**. 따라서 printStar에서는 전역 변수의 값 10이, main의 printf에서는 지역 변수의 값 5가 출력된다.

NoteBase

공통 해설

C에서는 변수를 사용하는 코드의 위치와 선언 범위에 따라 어떤 변수를 참조할지가 결정된다. 같은 이름의 전역 변수와 지역 변수가 있으면 **지역 변수의 유효 범위 안에서는 지역 변수가 우선**한다. 이 프로그램의 전역 변수 star는 10이고, main 안에 선언된 지역 변수 star는 5이며, 둘은 서로 다른 변수이다.

printStar의 printf에서 사용하는 star는 전역 변수이다. main의 지역 변수는 main의 블록 안에서만 유효하므로, main이 printStar를 호출해도 printStar 안에서 그 지역 변수를 사용할 수는 없다. 따라서 첫 번째 출력값은 10이다.

printStar가 끝난 뒤 main의 printf가 실행된다. 이 위치에서는 지역 변수 star가 전역 변수를 가리므로 두 번째 출력값은 5이다. 각 printf의 줄바꿈에 따라 10과 5가 차례로 서로 다른 줄에 출력된다.

NoteBase

문 20. 다음과 같이 P1, P2, P3, P4 프로세스가 동시에 준비 상태 큐에 도착했을 때 SJF(Shortest Job First) 스케줄링 알고리즘에서 평균 반환시간과 평균 대기시간을 바르게 연결한 것은? (단, 프로세스 간 문맥교환에 따른 오버헤드는 무시하며, 주어진 4개의 프로세스 외에 처리할 다른 프로세스는 없다고 가정한다)

프로세스	실행시간
P1	5
P2	6
P3	4
P4	9

① 평균 반환시간: 6, 평균 대기시간: 6

② 평균 반환시간: 6, 평균 대기시간: 7

③ 평균 반환시간: 13, 평균 대기시간: 6

④ 평균 반환시간: 13, 평균 대기시간: 7

정답 4번

간단한 해설

SJF는 **실행시간이 짧은 프로세스부터** 처리한다. 모두 동시에 도착하므로 완료될 때까지의 시간을 평균하면 반환시간은 13이고, 실행을 시작하기 전까지 기다린 시간을 평균하면 대기시간은 7이다.

NoteBase

계산 과정

실행 순서와 시각: 표의 실행시간은 P1이 5, P2가 6, P3가 4, P4가 9이다. SJF는 준비된 프로세스 중 **실행시간이 가장 짧은 것부터** 처리하므로 P3 → P1 → P2 → P4 순서로 실행한다. 동시에 도착한 시점을 0으로 두면, 문맥교환 오버헤드와 다른 프로세스가 없으므로 P3는 0~4, P1은 4~9, P2는 9~15, P4는 15~24에 실행된다.

평균 반환시간: 반환시간은 **도착부터 실행 완료까지 걸린 시간**으로, 완료시각에서 도착시각을 뺀 값이다. 모두 시각 0에 도착했으므로 P3, P1, P2, P4의 반환시간은 각각 4, 9, 15, 24이다. 네 프로세스의 평균 반환시간은 $\frac{4+9+15+24}{4} = 13$이다.

평균 대기시간: 이 문제에서 대기시간은 준비 상태 큐에서 **실행을 시작하기 전까지 기다린 시간**이다. P3는 즉시 실행하므로 0, P1은 P3의 실행이 끝날 때까지 4, P2는 4 + 5 = 9, P4는 4 + 5 + 6 = 15만큼 기다린다. 각 값은 해당 프로세스의 반환시간에서 자신의 실행시간을 뺀 값과도 같다. 평균 대기시간은 $\frac{0+4+9+15}{4} = 7$이다.

따라서 평균 반환시간 13과 평균 대기시간 7의 연결은 ④와 일치한다. 두 값의 시간 단위는 표에 주어진 실행시간의 단위와 같다.

NoteBase