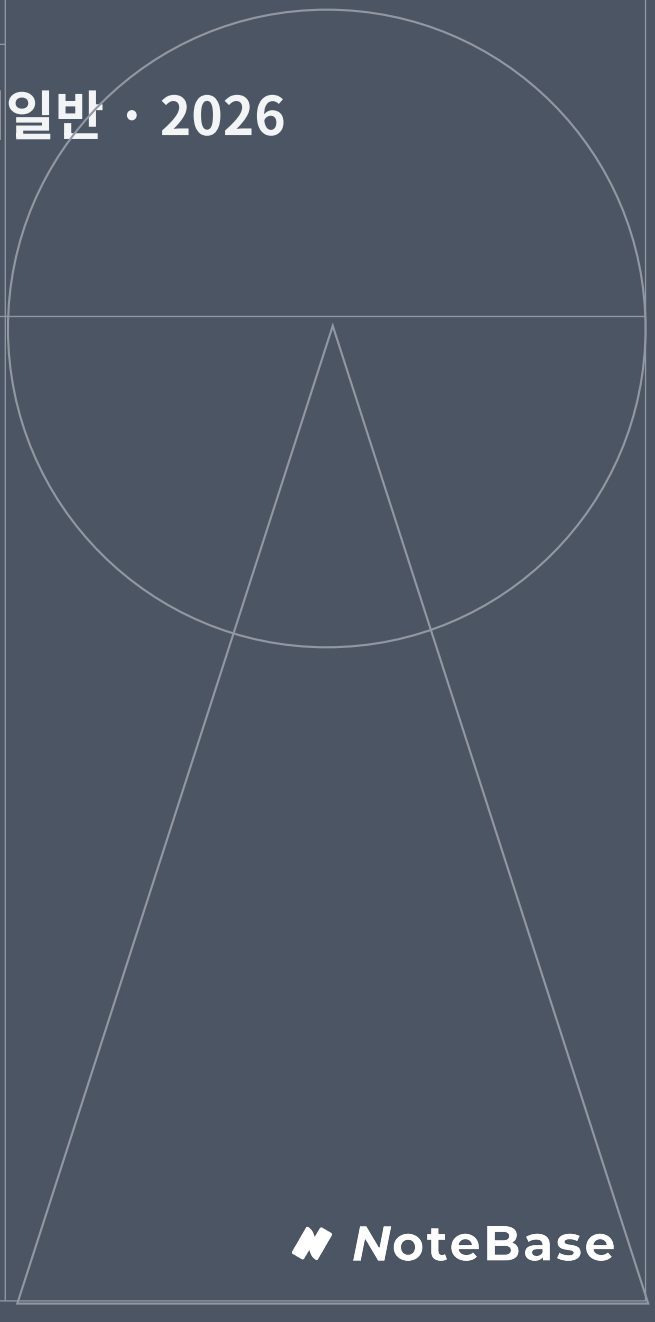
기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2026



NoteBase

기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2026

공무원/지방직/9급 · 컴퓨터일반 · 2026

문 1. 다음 설명에 해당하는 문자 데이터 표현은?

전 세계 언어를 하나의 코드 체계 안으로 통합하기 위해 만들어진 코드로서 2바이트를 사용해서 각 국가의 언어를 표현할 수 있다.

① Unicode
② ASCII 코드
③ BCD 코드
④ EBCDIC 코드

정답 1번

간단한 해설

여러 나라의 문자를 **하나의 공통 코드 체계로 통합**하려는 문자 표준이 Unicode이다. 지문의 '2바이트'는 초기 Unicode의 16비트 설계를 가리키는 설명이다.

핵심 풀이

Unicode는 한글·한자·라틴 문자 등 여러 언어의 문자에 공통된 체계로 고유한 코드 값을 부여하는 문자 표준이다. '전 세계 언어를 하나의 코드 체계 안으로 통합'한다는 목적이 **Unicode를 가리키는 결정적인 단서**이다.

ASCII 코드는 기본적으로 영문자·숫자·제어 문자 등을 표현하는 7비트 코드이다. 따라서 세계 여러 언어의 문자를 포괄하는 공통 체계라는 설명과 구별된다.

지문의 '2바이트'는 초기 Unicode가 16비트 체계로 설계되었다는 점에 해당한다. 다만 **Unicode 문자가 항상 2바이트인 것은 아니다**. UTF-16에서는 기본 다국어 평면에 속한 문자의 코드 포인트를 2바이트로, 보충 평면에 속한 문자의 코드 포인트를 4바이트로 표현한다.

NoteBase

문 2. TELNET보다 안전한 원격 접속 기능을 제공하는 서비스는?

① SSH(Secure SHell)

② VoIP(Voice over IP)

③ SMTP(Simple Mail Transfer Protocol)

④ DHCP(Dynamic Host Configuration Protocol)

정답 1번

간단한 해설

SSH는 원격 접속 시 로그인 정보와 명령·응답 데이터를 **암호화하여 전송**한다. 기본적인 TELNET의 평문 통신과 달리 통신 내용을 엿보는 도청 위험을 줄인다.

개념 구분

TELNET은 원격 컴퓨터에 접속하여 명령을 실행하는 기능을 제공하지만, 기본적으로 로그인 정보와 통신 내용을 **평문으로 전송**한다. 따라서 전송 중인 데이터를 가로채면 그 내용을 읽을 수 있다.

SSH(Secure Shell)는 **암호화된 통신 경로**를 통해 사용자 인증과 원격 명령 실행을 제공한다. TELNET과 같은 원격 접속 목적을 수행하면서 전송 내용의 기밀성과 무결성을 보호하므로, 문제에서 묻는 더 안전한 원격 접속 서비스에 해당한다.

NoteBase

문 3. MAC(Media Access Control) 주소를 표현하는 데 필요한 16진수 개수는?

① 4

② 8

③ 12

④ 16

정답 3번

간단한 해설

일반적인 MAC 주소는 48비트이며, **16진수 한 자리는 4비트**를 나타내므로 총 12자리로 표현한다.

공통 해설

일반적인 이더넷 MAC 주소의 길이는 48비트, 즉 6바이트이다. 16진수 한 자리는 4비트를 나타내므로 1바이트는 16진수 두 자리로 표현한다. 따라서 MAC 주소는 **두 자리씩 여섯 묶음인 총 12자리**의 16진수로 나타낸다.

예를 들어 00:1A:2B:3C:4D:5E는 16진수 두 자리로 된 묶음이 여섯 개이다. 묶음을 구분하는 콜론(:)은 16진수 자릿수에 포함하지 않는다.

NoteBase

문 4. 다음 설명에 해당하는 캐시(cache) 기억장치의 교체 알고리즘은?

캐시 안에 가장 오래된 블록을 교체하는 방식으로 쉽게 구현할 수 있으나 적중률 면에서는 높은 효율을 보이지 않는다.

① FIFO(First-In First-Out)
② LFU(Least Frequently Used)
③ LRU(Least Recently Used)
④ Random

정답 1번

간단한 해설

캐시에 들어온 시점이 가장 이른 블록을 먼저 교체하는 방식은 FIFO이다. 최근 사용 여부가 아니라 들어온 순서를 기준으로 삼는다.

핵심 풀이

FIFO(First-In First-Out)는 캐시에 먼저 들어온 블록을 먼저 내보내는 선입선출 방식이다. 지문의 '가장 오래된 블록'은 **캐시에 들어온 뒤 가장 오래 머문 블록**을 뜻한다. 들어온 순서를 큐로 관리하면 교체 대상을 쉽게 정할 수 있다. 다만 사용 빈도나 최근 사용 여부를 반영하지 않아 자주 사용하는 블록도 오래되었다는 이유로 교체될 수 있으므로, 높은 적중률을 보장하지 않는다.

LRU와의 구분: LRU는 **마지막으로 사용한 시점이 가장 오래된 블록**을 교체한다. 예를 들어 A, B, C가 차례로 캐시에 들어온 뒤 A를 다시 사용하고 새 블록 D를 넣기 위해 하나를 교체한다면, FIFO는 먼저 들어온 A를 내보낸다. 반면 LRU는 최근에 사용한 A를 남기고 가장 오랫동안 사용하지 않은 B를 내보낸다.

NoteBase

문 5. 10진수 +5를 1의 보수(1’s complement)로 변환하면? (단, 저장 공간은 8비트로 한다)

① 11111110

② 11111101

③ 11111010

④ 11111011

정답 3번

간단한 해설

+5를 8비트 이진수로 쓰면 00000101이다. 이 비트열의 **0과 1을 모두 반전**하면 1의 보수인 11111010이 된다.

공통 해설

주어진 비트열의 1의 보수는 **각 비트의 0을 1로, 1을 0으로 바꾼** 비트열이다. 저장 공간이 8비트이므로 +5의 이진수 101 앞에 0을 채워 00000101로 나타낸 뒤, 여덟 비트를 모두 반전한다.

00000101 → 11111010

따라서 +5의 8비트 비트열에 1의 보수 변환을 적용한 결과는 11111010이다. 비트를 반전한 뒤 1을 더하지는 않는다.

NoteBase

문 6. 순차 탐색(sequential search)과 이진 탐색(binary search)에 대한 설명으로 옳은 것은?

① 순차 탐색의 시간 복잡도는 '평균의 경우' $O(n^2)$이다.

② 이진 탐색은 재귀(recursive) 호출 방식으로 구현 가능하다.

③ 순차 탐색은 입력 데이터가 정렬되어 있을 때 원하는 값을 효율적으로 탐색하는 알고리즘이다.

④ 이진 탐색은 입력 데이터의 처음부터 끝까지 차례대로 비교하며 원하는 값을 탐색하는 알고리즘이다.

정답 2번

간단한 해설

이진 탐색은 정렬된 자료의 중앙값과 찾는 값을 비교하여 탐색 범위를 절반으로 줄인다. 줄어든 범위에 같은 탐색을 다시 적용하는 구조이므로 **재귀 호출**로 구현할 수 있다.

선지별 해설

① X: 순차 탐색은 자료를 하나씩 비교하므로 평균 시간 복잡도는 **자료 수 n에 비례하는** $O(n)$이다. 탐색이 성공하고 찾는 값이 각 위치에 있을 가능성이 같다면 평균적으로 전체 자료의 약 절반을 비교한다. 따라서 평균 시간 복잡도를 $O(n^2)$으로 제시한 부분을 $O(n)$으로 고친다.

② O: 이진 탐색은 정렬된 자료에서 중앙값과 찾는 값을 비교한 뒤, 찾는 값이 더 작으면 왼쪽 범위, 더 크면 오른쪽 범위를 탐색한다. 이때 **선택한 범위를 대상으로 자기 자신을 다시 호출**하도록 구현할 수 있다. 값을 찾거나 탐색 범위가 비면 재귀 호출을 종료한다.

③ X: 순차 탐색은 **자료가 정렬되어 있지 않아도** 앞에서부터 하나씩 비교하여 값을 찾을 수 있다. 정렬된 자료에서 중앙값과의 대소 비교로 탐색 범위를 절반씩 줄여 효율적으로 찾는 방식은 이진 탐색이다. 따라서 선지의 알고리즘 이름을 '이진 탐색'으로 고친다.

④ X: 처음부터 끝까지 차례대로 비교하는 방식은 **순차 탐색**이다. 이진 탐색은 정렬된 자료의 중앙값과 찾는 값을 비교하여, 값이 있을 수 없는 쪽을 제외하고 남은 절반을 탐색한다. 따라서 모든 자료를 순서대로 비교한다는 설명을 중앙값 비교로 탐색 범위를 절반씩 줄인다는 설명으로 고친다.

NoteBase

문 7. TCP/IP 프로토콜 계층 구조에서 응용 계층 프로토콜이 아닌 것은?

① FTP

② RIP

③ HTTP

④ SMTP

정답 2번

간단한 해설

RIP는 라우터 사이에서 **경로 정보를 교환하는 라우팅 프로토콜**이다. 이 문항에서는 그 기능을 기준으로 파일 전송·웹·전자우편 같은 응용 서비스를 제공하는 프로토콜과 구분한다.

선지별 해설

① O: FTP는 클라이언트와 서버 사이에서 **파일을 업로드하거나 다운로드**하는 응용 계층 프로토콜이다.

② X: RIP는 라우터 사이에서 경로 정보를 교환하여 라우팅 테이블을 갱신하는 프로토콜이다. 이 문항의 기능 중심 분류에서는 응용 서비스가 아니라 **인터넷 계층의 경로 설정을 지원하는 프로토콜**로 구분한다. RIP 메시지 자체는 UDP를 이용해 전달되지만, 수행하는 기능은 패킷 전달에 필요한 경로 정보를 제공하는 것이다.

③ O: HTTP는 웹 클라이언트와 서버 사이에서 **요청과 응답**을 주고받는 응용 계층 프로토콜이다. 웹 브라우저가 웹 문서를 요청하고 서버가 해당 내용을 보내는 통신에 사용한다.

④ O: SMTP는 전자우편을 **발송하고 메일 서버 사이에서 전달**하는 응용 계층 프로토콜이다. 수신함에 저장된 메일을 사용자가 조회하는 POP3·IMAP과는 역할이 다르다.

NoteBase

문 8. 관계형 데이터 모델에서 릴레이션(relation)에 대한 설명으로 옳은 것은?

① 한 릴레이션에서 튜플(tuple)의 중복을 허용한다.

② 속성(attribute)은 원자 값(atomic value)을 가진다.

③ 한 릴레이션에 포함된 튜플 간의 순서는 의미가 있다.

④ 한 릴레이션을 구성하는 속성 간의 순서는 의미가 있다.

정답 2번

간단한 해설

릴레이션의 각 튜플에서 속성 값은 **하나의 원자 값**이어야 한다. 이는 한 칸에 여러 값의 집합을 넣지 않고, 더 나누지 않는 하나의 값으로 취급한다는 뜻이다.

선지별 해설

릴레이션은 표로 표현할 수 있으며, 행은 튜플, 열은 속성에 해당한다.

① X: 릴레이션은 튜플의 집합이므로 **동일한 튜플의 중복을 허용하지 않는다**. 두 튜플의 모든 속성 값이 같으면 같은 튜플이며, 일부 속성 값만 같은 것은 허용된다.

② O: 한 튜플의 각 속성에는 **하나의 원자 값**이 들어간다. 원자 값은 해당 데이터 모델에서 더 나누지 않는 하나의 값으로 취급하는 값이다. 한 칸에 여러 값의 집합이나 반복 그룹을 두지 않는 이 성질은 제1정규형의 기준이다.

③ X: 릴레이션은 집합이므로 **튜플 간의 순서는 의미가 없다**. 같은 튜플들을 표의 위아래 어느 순서로 배치하더라도 릴레이션이 나타내는 정보는 같다.

④ X: 릴레이션에서 **속성 간의 순서는 의미가 없다**. 속성은 열의 위치가 아니라 속성 이름으로 구별한다. 예를 들어 학번과 이름 열의 위치를 바꾸더라도 각 속성에 대응하는 값을 함께 옮기면 같은 정보를 나타낸다.

NoteBase

문 9. 중앙처리장치의 레지스터(register)가 아닌 것은?

① MBR(Memory Buffer Register)

② PC(Program Counter)

③ IR(Instruction Register)

④ ALU(Arithmetic and Logical Unit)

정답 4번

간단한 해설

ALU는 덧셈·뺄셈 같은 산술 연산과 AND·OR 같은 논리 연산을 수행하는 **연산장치**이다. 정보를 임시로 저장하는 레지스터와는 역할이 다르다.

선지별 해설

레지스터는 중앙처리장치 내부에서 명령어·주소·데이터 등을 일시적으로 보관하는 고속 저장장치이다. 각 구성 요소가 정보를 저장하는지, 연산을 수행하는지를 구분한다.

① O: MBR(메모리 버퍼 레지스터)은 **메모리에서 읽어 온 내용이나 메모리에 쓸 내용**을 임시로 저장하는 레지스터이다. 메모리와 중앙처리장치 사이에서 전달되는 명령어나 데이터를 보관한다.

② O: PC(프로그램 카운터)는 **다음에 인출할 명령어의 주소**를 저장하는 레지스터이다. 명령어의 내용 자체가 아니라 그 명령어가 있는 위치를 기억한다.

③ O: IR(명령어 레지스터)은 메모리에서 인출하여 **현재 해독하거나 실행할 명령어**를 저장하는 레지스터이다. PC가 명령어의 주소를 보관하는 것과 구별된다.

④ X: ALU(산술논리연산장치)는 데이터를 저장하는 레지스터가 아니라 **산술 연산과 논리 연산을 수행하는 회로**이다. 예를 들어 두 수를 더하는 처리는 ALU가 수행하며, 연산에 필요한 값이나 결과를 보관하는 역할은 레지스터가 맡는다.

NoteBase

문 10. OSI 7 계층과 계층별로 사용하는 네트워크 장비를 옳게 짝지은 것은?

① 전송 계층 – 허브(hub)

② 데이터 링크 계층 – 라우터(router)

③ 데이터 링크 계층 – 브릿지(bridge)

④ 데이터 링크 계층 – 리피터(repeater)

정답 3번

간단한 해설

브릿지는 프레임의 **MAC 주소**를 확인하여 LAN 구간 사이의 전달 여부를 결정하므로 **데이터 링크 계층**에 해당한다.

선지별 해설

① X: 허브는 수신한 신호를 다른 포트로 전달하는 **물리 계층** 장비이다. 주소를 해석하여 목적지를 선택하지 않으므로 전송 계층과의 연결은 잘못되었다.

② X: 라우터는 패킷의 목적지 IP 주소와 라우팅 정보를 바탕으로 다음 전달 경로를 결정하는 **네트워크 계층** 장비이다. 따라서 데이터 링크 계층이 아니라 네트워크 계층과 짝지어야 한다.

③ O: 브릿지는 LAN 구간을 연결하며, 프레임의 **MAC 주소를 기준으로 전달 여부를 결정**한다. 프레임과 MAC 주소를 다루는 데이터 링크 계층의 장비이므로 연결이 맞는다.

④ X: 리피터는 전송 중 약해진 신호를 재생하여 더 멀리 전달하는 **물리 계층** 장비이다. 프레임의 주소를 해석하지 않으므로 데이터 링크 계층이 아니라 물리 계층과 짝지어야 한다.

NoteBase

문 11. 기계 학습의 학습 유형인 비지도 학습(unsupervised learning)에 대한 설명으로 옳은 것은?

① 핵심 개념은 에이전트, 환경, 상태, 액션, 보상 등이다.

② 군집화(clustering)를 사용할 수 없다.

③ 학습 알고리즘이 누적 보상을 최대화하는 방향으로 진행된다.

④ 레이블이 없는 학습용 데이터만을 대상으로 학습 알고리즘이 스스로 패턴을 발견한다.

정답 4번

간단한 해설

비지도 학습은 **레이블이 없는 데이터**에서 유사성이나 분포 같은 **숨은 구조와 패턴**을 스스로 찾아내는 학습이다. 미리 지정된 범주나 목표값을 제공하지 않는다는 점이 핵심이다.

선지별 해설

① X: 에이전트, 환경, 상태, 액션, 보상은 **강화 학습**의 핵심 개념이다. 에이전트가 환경의 상태를 관찰하고 액션을 선택하면, 환경으로부터 보상과 다음 상태를 받아 행동 방식을 학습한다. 비지도 학습은 이러한 보상에 따른 행동 학습이 아니라 레이블이 없는 데이터의 구조를 찾는 방식이다.

② X: 군집화는 비지도 학습의 **대표적인 방법**이다. 미리 범주 레이블을 주지 않고 데이터 간 유사성을 기준으로 비슷한 데이터끼리 묶는다. 따라서 '군집화를 사용할 수 없다'는 '군집화를 사용할 수 있다'로 고쳐야 한다.

③ X: **누적 보상을 최대화**하도록 행동 방식을 학습하는 것은 강화 학습이다. 당장의 보상뿐 아니라 이후에 받을 보상까지 고려하여 행동을 선택하도록 학습한다. 비지도 학습은 누적 보상 대신 데이터 내부의 유사성이나 분포 등 구조와 패턴을 찾는다.

④ O: 비지도 학습에서는 학습용 데이터에 **목표값이나 범주를 나타내는 레이블을 붙이지 않는다**. 알고리즘은 데이터 자체의 특성과 관계를 바탕으로 패턴을 찾는다. 예를 들어 고객별 구매 기록에 고객 유형을 미리 표시하지 않고, 구매 특성이 비슷한 고객들을 묶는 군집화가 이에 해당한다.

NoteBase

문 12. 다음 C 프로그램의 출력 결과는?

```
#include <stdio.h>
#include <string.h>
int main(void) {
    char s[] = "COMPUTER";
    char* p = s + 2;
    *(p + 3) = 'X';
    printf("%s\n", s);
    printf("%zu\n", strlen(p));
    return 0;
}
```

① COMXUTER
5
② COMPUXER
5
③ COMXUTER
6
④ COMPUXER
6

정답 4번

간단한 해설

p + 3은 **s[5]**를 가리키므로 'T'가 'X'로 바뀌어 COMPUXER가 출력된다. p는 여전히 s[2]를 가리키므로 **strlen(p)는 6**이다.

NoteBase

계산 과정

1. 포인터의 시작 위치: 배열의 인덱스는 0부터 시작한다. s에는 'COMPUTER'의 문자 8개와 문자열 끝을 나타내는 널 문자 '\0'이 저장된다. p = s + 2이므로 p는 **s[2]의 'M'**을 가리키며, s[8]에는 널 문자가 있다.

2. 배열 원소 변경과 첫 출력: p + 3은 s의 시작에서 2 + 3 = 5칸 떨어진 s[5]의 주소이다. *(p + 3) = 'X'는 **그 주소에 저장된 문자**를 바꾸므로 s[5]의 'T'가 'X'로 바뀐다. 이 연산으로 p 자체가 이동하지는 않는다. 첫 printf는 s의 시작부터 문자열을 출력하므로 첫 줄은 COMPUXER이다.

3. 문자열 길이와 둘째 출력: strlen(p)는 **p가 가리키는 위치부터 널 문자 직전까지**의 문자 수를 센다. p가 가리키는 s[2]부터 읽으면 'MPUXER'이므로 문자 수는 8 − 2 = 6이다. 'T'를 'X'로 바꾸어도 널 문자의 위치는 변하지 않으므로 둘째 줄에는 6이 출력된다.

따라서 첫 줄에 COMPUXER, 둘째 줄에 6이 출력되며, 이는 ④의 출력 결과와 일치한다.

NoteBase

문 13. (가), (나)에 들어갈 용어를 바르게 연결한 것은?

(가) 시스템 호출은 실행 중인 프로세스로부터 새로운 프로세스를 복사하는 함수로 기존 프로세스는 부모 프로세스, 새로 생성된 프로세스는 자식 프로세스가 된다. (가)를 이용해 생성된 자식 프로세스에게는 (나)을 반환한다.

① (가): exec(), (나): 0

② (가): exec(), (나): 1

③ (가): fork(), (나): 0

④ (가): fork(), (나): 1

정답 3번

간단한 해설

실행 중인 프로세스를 복제해 자식 프로세스를 만드는 호출은 **fork()**이다. 생성에 성공하면 자식 프로세스에서는 **0**을 반환한다.

빈칸별 해설

(가) fork(): 실행 중인 부모 프로세스를 복제하여 새로운 자식 프로세스를 만든다. 부모와 자식은 각각 fork() 호출 다음 지점부터 실행을 이어 간다. 반면 exec() 계열 함수는 **현재 프로세스의 실행 내용을 새 프로그램으로 교체**하므로, 새로운 프로세스를 복제하는 호출에 해당하지 않는다.

(나) 0: fork()가 성공하면 **자식 프로세스에서는 0, 부모 프로세스에서는 자식의 프로세스 ID(PID)**를 반환한다. 두 프로세스는 이 반환값으로 자신이 부모인지 자식인지 구분할 수 있다. 자식에게 반환되는 0은 자식의 PID가 0이라는 뜻이 아니며, 부모에게 반환되는 값도 항상 1인 것은 아니다.

따라서 (가)는 fork(), (나)는 0으로 연결되며, 이 조합은 ③과 일치한다.

NoteBase

문 14. 스택(stack)과 큐(queue)에 대한 설명으로 옳은 것은?

① 스택은 먼저 삽입된 데이터가 먼저 삭제되는 FIFO(First-In First-Out) 구조이다.

② 큐는 나중에 삽입된 데이터가 먼저 삭제되는 LIFO(Last-In First-Out) 구조이다.

③ 시스템 스택은 프로그램 실행 시 함수의 호출과 복귀에 따른 실행 순서를 관리할 수 있다.

④ 큐는 배열(array)을 사용하여 구현할 수 없다.

정답 3번

간단한 해설

함수 호출 시 복귀할 위치와 실행에 필요한 정보를 시스템 스택에 저장하고, 함수가 끝나면 꺼내어 호출한 곳으로 돌아간다. **나중에 호출된 함수부터 먼저 복귀**하는 관계가 스택의 후입선출 구조와 맞는다.

선지별 해설

① X: 스택은 **나중에 삽입된 데이터가 먼저 삭제되는 LIFO(후입선출)** 구조이다. 삽입과 삭제가 모두 같은 쪽 끝인 top에서 이루어진다. 예를 들어 A를 넣고 B를 넣으면 B가 먼저 나온다.

② X: 큐는 **먼저 삽입된 데이터가 먼저 삭제되는 FIFO(선입선출)** 구조이다. 뒤쪽인 rear로 삽입하고 앞쪽인 front에서 삭제한다. A를 넣고 B를 넣으면 A가 먼저 나온다.

③ O: 시스템 스택은 함수가 호출될 때 복귀 주소와 지역 변수 등 실행에 필요한 정보를 저장하고, 함수가 종료되면 해당 정보를 꺼내 호출한 함수의 실행을 이어 가게 한다. 예를 들어 A가 B를 호출하고 B가 C를 호출하면, C가 끝나 B로 돌아간 뒤 B가 끝나 A로 돌아간다. 이처럼 **가장 최근의 호출부터 복귀**하므로 후입선출 구조로 호출과 복귀에 따른 실행 순서를 관리할 수 있다.

④ X: 큐는 **배열이나 연결 리스트로 구현할 수 있다**. 배열로 구현할 때는 front와 rear 인덱스로 삭제할 위치와 삽입할 위치를 관리한다. 배열의 끝에 도달한 뒤 앞쪽의 빈 공간을 다시 활용하도록 원형 큐로 구현할 수도 있다.

NoteBase

문 15. 다음에서 설명하는 시스템 버스(bus) 중재 방식은?

◯ 버스 중재기가 각 버스 마스터들이 버스 사용을 요구하는지 주기적으로 검사하여 버스 사용 여부를 결정한다.
◯ 하드웨어 방식과 소프트웨어 방식으로 구분된다.

① 병렬
② 직렬
③ 분산
④ 폴링

정답 4번

간단한 해설

버스 중재기가 각 마스터의 사용 요구를 **주기적으로 확인**하여 사용권을 주는 방식은 폴링이다. 이러한 확인 절차는 하드웨어 회로나 소프트웨어로 구현할 수 있다.

핵심 풀이

폴링의 동작: 버스 마스터는 버스를 사용하여 전송을 주도할 수 있는 장치이며, 버스 중재는 여러 마스터 중 버스 사용권을 가질 장치를 결정하는 과정이다. 폴링에서는 중재기가 **각 마스터를 차례로 지정하여 사용 요구를 확인**한다. 요구가 확인된 마스터에게 사용권을 부여하므로, 지문의 '주기적으로 검사'한다는 설명이 폴링을 가리킨다.

구현 방식: 요구 여부를 확인하는 절차를 전용 회로로 수행하면 하드웨어 폴링, 프로그램의 명령으로 수행하면 소프트웨어 폴링이다. 따라서 '하드웨어 방식과 소프트웨어 방식으로 구분된다'는 두 번째 설명도 폴링에 해당한다.

직렬 중재와의 구분: 직렬 중재의 대표적인 데이지 체인 방식은 **버스 승인 신호를 장치들의 연결 순서에 따라 전달**한다. 앞쪽 장치가 사용을 요구하면 승인 신호를 받아 버스를 사용한다. 폴링도 차례로 확인하지만, 구별 기준은 단순한 순차성이 아니라 중재기가 각 마스터의 요구 여부를 직접 확인한다는 점이다.

NoteBase

문 16. 프로세스 도착 시간과 실행 시간은 다음과 같다. FCFS(First-Come First-Served) 스케줄링 알고리즘을 사용하였을 경우, P_1, P_2, P_3, P_4의 반환 시간에 대한 평균은?

프로세스	도착 시간	실행 시간
P_1	0	3
P_2	2	8
P_3	5	4
P_4	8	3

① 6

② 8

③ 10

④ 12

정답 2번

간단한 해설

반환 시간은 **완료 시각에서 도착 시각을 뺀 값**으로, 대기 시간과 실행 시간을 모두 포함한다. FCFS로 도착 순서대로 실행하면 네 프로세스의 평균 반환 시간은 8이다.

NoteBase

계산 과정

FCFS는 **먼저 도착한 프로세스부터 끝까지 실행**하는 비선점 스케줄링 방식이다. 표의 도착 시간에 따라 $P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4$ 순서로 실행한다. 반환 시간은 프로세스가 도착한 때부터 실행을 마칠 때까지 걸린 시간이므로, 완료 시각에서 도착 시각을 뺀다. 시간 단위는 표에 제시된 것과 동일하게 사용한다.

P_1: 도착 시각은 0이고 실행 시간은 3이다. 시각 0부터 3까지 실행하므로 완료 시각은 3, 반환 시간은 $3 - 0 = 3$이다.

P_2: 시각 2에 도착하지만 앞선 프로세스가 끝나는 시각 3부터 실행한다. 실행 시간이 8이므로 완료 시각은 $3 + 8 = 11$, 반환 시간은 $11 - 2 = 9$이다. 이때 9는 **대기 시간 1과 실행 시간 8을 합한 값**이다.

P_3: 시각 5에 도착하여 대기하다가 시각 11부터 4만큼 실행한다. 완료 시각은 $11 + 4 = 15$, 반환 시간은 $15 - 5 = 10$이다.

P_4: 시각 8에 도착하여 대기하다가 시각 15부터 3만큼 실행한다. 완료 시각은 $15 + 3 = 18$, 반환 시간은 $18 - 8 = 10$이다.

평균 반환 시간은 네 반환 시간의 합을 프로세스 수 4로 나눈 $\frac{3+9+10+10}{4} = 8$이다. 따라서 선택지 ②의 8에 해당한다.

NoteBase

문 17. 딥러닝에 대한 설명으로 옳지 않은 것은?

① 딥러닝 모델은 2개 이상의 은닉층을 가질 수 있으며 음성 인식, 자연어 처리 등에 활용된다.

② 합성곱 계층(convolutional layer)은 이미지에 대한 특징 정보를 추출한다.

③ LSTM(Long Short-Term Memory)은 장기 및 단기 메모리 기능을 포함하는 합성곱 신경망(convolutional neural network)이다.

④ 순환 신경망(recurrent neural network)은 시간적으로 연속성이 있는 데이터를 처리하기 위한 인공 신경망이다.

정답 3번

간단한 해설

LSTM은 합성곱 신경망이 아니라 **순환 신경망(RNN)의 한 종류**이다. 메모리 셀과 게이트를 이용해 이전 정보를 유지하거나 버리면서 긴 시점 간의 의존 관계를 학습한다.

선지별 해설

① O: 딥러닝 모델은 **여러 은닉층을 쌓아** 입력의 특징을 단계적으로 학습할 수 있다. 따라서 은닉층이 2개 이상인 모델을 구성할 수 있으며, 음성의 특징을 학습하는 음성 인식이나 문맥을 학습하는 자연어 처리 등에 활용된다.

② O: 합성곱 계층은 학습된 필터를 이미지의 여러 위치에 적용하여 **국소적인 특징을 추출**한다. 예를 들어 선이나 경계 같은 특징을 검출하고, 계층을 쌓으면 이를 조합한 더 복잡한 형태의 특징을 학습할 수 있다.

③ X: LSTM은 **합성곱 신경망이 아니라 순환 신경망의 한 종류**이다. 메모리 셀에 정보를 저장하고 입력·망각·출력 게이트로 정보의 저장, 유지, 출력 정도를 조절하여 멀리 떨어진 시점의 정보도 활용하도록 한다. 합성곱 신경망이 필터로 공간적 특징을 추출하는 것과 달리, LSTM은 순차 데이터의 시간적 의존 관계를 다루는 구조이다.

④ O: 순환 신경망은 **이전 시점의 은닉 상태를 현재 시점의 처리에 반영**하여 순서에 따른 정보를 학습한다. 따라서 시계열·음성처럼 시간적으로 연속성이 있는 데이터를 처리하는 데 적합하며, 단어가 차례로 주어지는 문장과 같은 순차 데이터에도 활용된다.

NoteBase

문 18. 다음 Java 프로그램의 실행 결과는?

```
class Main {
   public static void div(double a, double b){
       try{
            System.out.println(a / b);
       }catch(ArithmeticException e){
            System.out.println("divide by zero");
       }
   }
   public static void main(String args[]) {
       div(3, 2);
       div(2, 0);
   }
}
```

① 1
Infinity
② 1.5
Infinity
③ 1
divide by zero
④ 1.5
divide by zero

정답 2번

간단한 해설

매개변수가 **double형**이므로 첫 나눗셈은 소수 부분을 유지하여 1.5를 출력한다. 두 번째는 양의 실수를 0.0으로 나누는 부동소수점 연산이므로 예외 없이 **Infinity**를 출력한다.

NoteBase

계산 과정

연산 자료형: div의 매개변수 a와 b는 모두 double형이다. 호출할 때 전달한 정수는 double형으로 변환되므로, a / b는 **부동소수점 나눗셈**으로 처리된다. 정수끼리 나눌 때처럼 소수 부분을 버리지 않는다.

div(3, 2): a는 3.0, b는 2.0이 된다. 따라서 $\frac{3.0}{2.0} = 1.5$이며, 첫 번째 println은 1.5를 출력한다.

div(2, 0): a는 2.0, b는 0.0이 된다. Java에서 양의 유한한 부동소수점 값을 양의 0.0으로 나누면 양의 무한대가 되어 Infinity를 출력한다. **부동소수점 나눗셈은 0으로 나누어도 ArithmeticException을 발생시키지 않으므로** catch 블록의 'divide by zero'는 출력되지 않는다. 정수 나눗셈에서 0으로 나누는 경우와 구별한다.

두 호출은 main에 적힌 순서대로 실행되고 println은 각각 줄을 바꾸므로, 첫째 줄에 1.5, 둘째 줄에 Infinity가 출력된다. 이는 ②의 실행 결과와 일치한다.

NoteBase

문 19. 다중 스레드(multithreading) 모델에 대한 설명으로 옳은 것만을 모두 고른 것은?

ㄱ. 스레드는 사용자 스레드와 커널 스레드가 있다.
ㄴ. 일대일 모델은 다중 처리기에서 다중 스레드가 병렬로 수행되는 것을 허용하지 않는다.
ㄷ. 다대일 모델에서는 프로세스에 속한 하나의 스레드가 입출력 작업을 위해 대기 상태에 들어가도 나머지 스레드는 계속 실행된다.

① ㄱ
② ㄱ, ㄴ
③ ㄱ, ㄷ
④ ㄴ, ㄷ

정답 1번

간단한 해설

일대일 모델은 사용자 스레드마다 별도의 커널 스레드를 연결하므로 **다중 처리기에서 병렬 실행이 가능하다**. 다대일 모델은 여러 사용자 스레드가 하나의 커널 스레드를 공유하므로, 블로킹 입출력으로 그 커널 스레드가 대기하면 **다른 스레드도 실행할 수 없다**.

보기별 해설

ㄱ. O: 스레드는 사용자 수준에서 스레드 라이브러리가 관리하는 사용자 스레드와 운영체제 커널이 직접 관리하는 커널 스레드로 구분한다. 다중 스레드 모델은 이러한 **사용자 스레드와 커널 스레드의 대응 관계**를 나타낸다.

ㄴ. X: 일대일 모델은 사용자 스레드 하나를 커널 스레드 하나에 대응시킨다. 커널이 각 스레드를 독립적으로 스케줄링할 수 있으므로, 다중 처리기에서는 서로 다른 스레드를 각 처리기에 배치하여 **병렬로 실행할 수 있다**. 따라서 병렬 수행을 허용하지 않는다는 설명은 반대이다.

ㄷ. X: 다대일 모델은 여러 사용자 스레드를 **하나의 커널 스레드**에 대응시킨다. 한 사용자 스레드가 블로킹 입출력을 요청하여 이 커널 스레드가 대기 상태가 되면, 나머지 사용자 스레드도 실행에 필요한 커널 스레드를 사용할 수 없다. 따라서 나머지 스레드가 계속 실행되는 것이 아니라 프로세스 전체의 실행이 막힌다.

옳은 보기는 ㄱ뿐이므로, ㄱ만 제시한 ①에 해당한다.

NoteBase

문 20. 다음 설명에 해당하는 디자인 패턴은?

◯ 객체를 오직 하나만 생성한다.
◯ 외부에서 접근하는 것을 통제하기 위해 생성자를 private으로 선언한다.

① Adapter 패턴
② Composite 패턴
③ Singleton 패턴
④ Factory method 패턴

정답 3번

간단한 해설

Singleton 패턴은 클래스의 인스턴스를 하나만 생성하여 공유하는 패턴이다. 생성자를 private으로 선언하면 외부에서 생성자를 직접 호출해 새 인스턴스를 만드는 것을 막을 수 있다.

핵심 풀이

'객체를 오직 하나만 생성한다'는 단서는 **Singleton 패턴**을 가리킨다. 인스턴스는 클래스로부터 생성된 객체를 뜻하며, 이 패턴에서는 여러 곳에서 객체가 필요해도 하나의 인스턴스를 함께 사용한다.

'생성자를 private으로 선언한다'는 것은 **클래스 외부에서 생성자를 직접 호출할 수 없게 한다**는 뜻이다. 다만 private 선언만으로 인스턴스가 하나로 제한되는 것은 아니다. 클래스 내부에서 하나의 인스턴스를 관리하고, 외부에는 그 인스턴스를 얻는 접근 메서드를 제공하여 객체 생성을 통제한다.

Factory method 패턴은 객체 생성에 사용할 메서드를 정의하고, 구체적으로 어떤 클래스의 객체를 만들지는 하위 클래스에서 결정하게 한다. 핵심은 **생성할 객체의 종류를 결정하는 책임**이며, 인스턴스를 하나로 제한하는 Singleton 패턴과 구별된다.

NoteBase