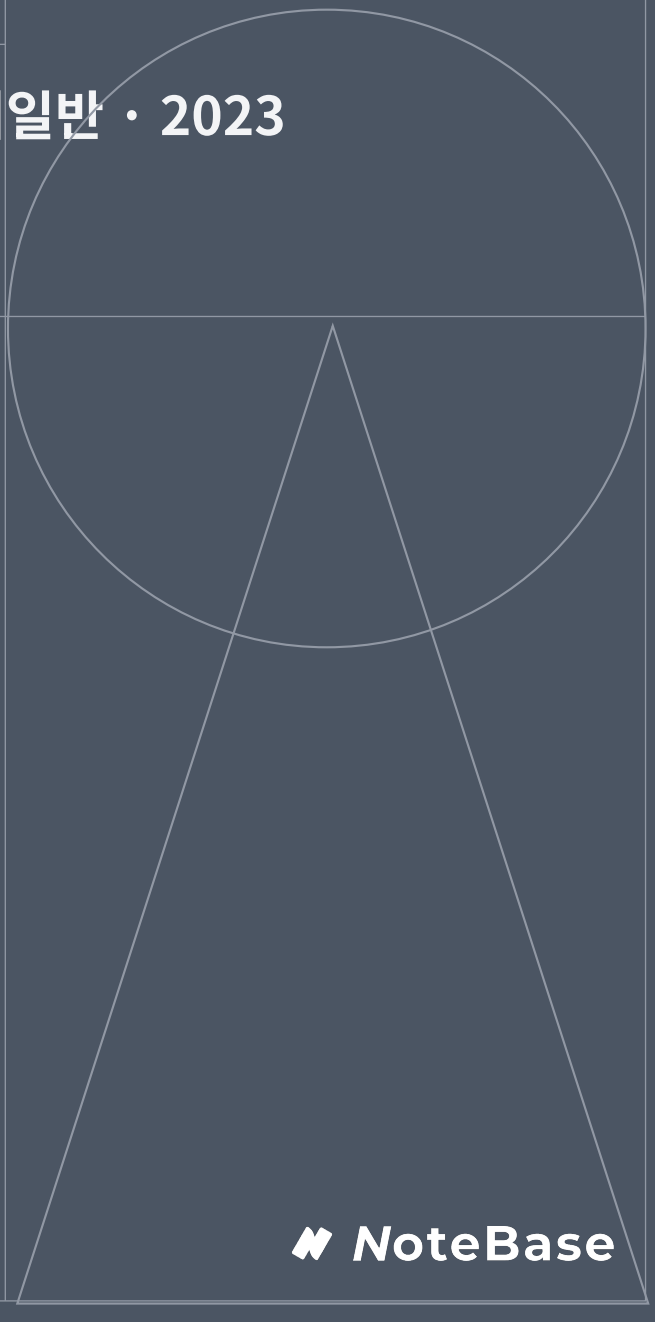
기출OX 해설집

공무원/국가직/9급 · 컴퓨터일반 · 2023



NoteBase

기출OX 해설집

공무원/국가직/9급 · 컴퓨터일반 · 2023

공무원/국가직/9급 · 컴퓨터일반 · 2023

문 1. 병렬 처리를 수행하는 기법으로 옳지 않은 것은?

① 블루-레이 디스크

② VLIW

③ 파이프라인

④ 슈퍼스칼라

정답 1번

간단한 해설

블루-레이 디스크는 데이터를 저장하는 **광학 저장 매체**이다. 여러 명령이나 연산을 겹치거나 동시에 수행하는 병렬 처리 기법에 해당하지 않는다.

선지별 해설

① X: 블루-레이 디스크는 레이저를 이용해 데이터를 읽고 쓰는 **광학 저장 매체**이다. 데이터 저장을 위한 매체이므로, 프로세서가 명령이나 연산을 병렬로 처리하는 기법으로 분류할 수 없다.

② O: VLIW는 **컴파일러가 병렬 실행 가능한 연산들을 하나의 긴 명령어로 묶는** 방식이다. 프로세서는 그 안의 연산들을 여러 실행 장치에서 동시에 수행하여 명령어 수준의 병렬성을 활용한다.

③ O: 파이프라인은 명령어 처리를 인출·해독·실행 등의 단계로 나누고, **서로 다른 명령어의 처리 단계를 겹쳐 수행**한다. 예를 들어 한 명령어를 실행하는 동안 다음 명령어를 해독할 수 있다. 이는 여러 명령어의 처리가 겹치도록 하여 단위 시간에 처리하는 명령어 수를 늘리는 방식이다.

④ O: 슈퍼스칼라는 여러 실행 장치를 이용하여 **한 클록 주기에 여러 명령어를 발행**하고 병렬로 실행할 수 있는 방식이다. 하드웨어가 명령어 간 의존성과 실행 장치의 사용 가능 여부 등을 판단하여 함께 실행할 명령어를 정한다. 컴파일러가 연산을 미리 묶는 VLIW와 이 점에서 구별된다.

NoteBase

문 2. 인터넷 통신에서 IP 주소를 동적으로 할당하는 데 사용되는 것은?

① TCP

② DNS

③ SOAP

④ DHCP

정답 4번

간단한 해설

DHCP는 컴퓨터가 네트워크에 접속할 때 **IP 주소를 동적으로 할당**하는 프로토콜이다. 사용자가 각 컴퓨터의 IP 주소를 일일이 설정하지 않아도 되게 한다.

개념 구분

DHCP(Dynamic Host Configuration Protocol)는 호스트에 IP 주소와 통신에 필요한 설정 정보를 자동으로 제공하는 프로토콜이다. 컴퓨터가 주소를 요청하면 DHCP 서버는 사용 가능한 IP 주소를 **일정한 임대 기간 동안 할당**한다. 이처럼 필요한 때 주소를 받아 사용하는 방식이 문제에서 말하는 동적 할당이다.

DNS는 **도메인 이름에 대응하는 IP 주소를 조회**하는 데 사용된다. 접속할 상대의 주소를 알아내는 것과 자신의 컴퓨터에 사용할 주소를 할당받는 것은 다르므로, IP 주소의 동적 할당은 DHCP의 역할이다.

NoteBase

문 3. UDP 프로토콜에 대한 설명으로 옳지 않은 것은?

① 흐름 제어가 필요없는 비신뢰적 통신에 사용한다.

② 순차적인 데이터 전송을 통해 전송을 보장한다.

③ 비연결지향으로 송신자와 수신자 사이에 연결 설정 없이 데이터 전송이 가능하다.

④ 전송되는 데이터 중 일부가 손실되는 경우 손실 데이터에 대한 재전송을 요구하지 않는다.

정답 2번

간단한 해설

UDP는 데이터가 **보낸 순서대로 도착하거나 반드시 전달되는 것을 보장하지 않는다**. 순서 제어와 재전송으로 신뢰성 있는 전송을 제공하는 것은 TCP이다.

선지별 해설

① O: 흐름 제어는 수신자가 처리할 수 있는 속도에 맞추어 송신량을 조절하는 기능이다. UDP 자체는 **흐름 제어와 신뢰성 보장 기능을 제공하지 않으므로**, 일부 데이터 손실을 감수할 수 있고 이러한 기능이 필요하지 않은 통신에 사용할 수 있다.

② X: UDP는 각 데이터그램을 독립적으로 보내므로 **전달 순서와 전달 성공을 보장하지 않는다**. 예를 들어 A를 보낸 뒤 B를 보내도 B가 먼저 도착하거나 A가 손실될 수 있다. 선지의 설명은 TCP의 특성에 해당한다. TCP는 순서 번호로 데이터 순서를 관리하고 확인 응답과 재전송으로 신뢰성 있는 전송을 제공한다.

③ O: UDP는 데이터를 보내기 전에 상대와 논리적인 연결을 설정하는 절차를 거치지 않는 **비연결형 프로토콜**이다. 따라서 송신자는 수신자와 사전 연결 설정 없이 데이터그램을 전송할 수 있다.

④ O: UDP 자체에는 데이터의 수신 여부를 확인하고 손실된 데이터를 다시 보내도록 요구하는 기능이 없다. 따라서 데이터가 손실되어도 **UDP 차원에서는 재전송을 요구하지 않는다**. 필요하다면 UDP를 사용하는 응용 프로그램이 별도의 확인·재전송 기능을 구현해야 한다.

NoteBase

문 4. 플린(Flynn)의 분류법에 따른 병렬 프로세서 구조 중 MIMD(Multiple Instruction stream, Multiple Data stream) 방식에 속하지 않는 것은?

① 클러스터

② 대칭형 다중 프로세서

③ 불균일 기억장치 액세스

④ 배열 프로세서

정답 4번

간단한 해설

배열 프로세서는 **하나의 명령어 흐름으로 여러 데이터를 처리**하는 SIMD 방식이다. MIMD는 각 프로세서가 독립적인 명령어 흐름을 실행할 수 있다는 점에서 구별된다.

선지별 해설

MIMD는 여러 프로세서가 각자의 명령어 흐름으로 서로 다른 데이터를 처리하는 구조이다. 플린의 분류 기준은 명령어 흐름과 데이터 흐름의 수이며, 메모리를 공유하는지 여부는 별도의 구분 기준이다.

① O: 클러스터는 독립된 여러 컴퓨터를 네트워크로 연결하여 함께 작업하게 하는 구조이다. 각 컴퓨터가 **독립적인 명령어 흐름**으로 서로 다른 데이터를 처리할 수 있으므로 MIMD에 속한다.

② O: 대칭형 다중 프로세서(SMP)는 여러 프로세서가 주기억장치를 공유하는 구조이다. 메모리를 공유하더라도 각 프로세서는 자신의 명령어 흐름을 독립적으로 실행할 수 있으므로 MIMD에 속한다. **메모리 공유가 동일한 명령어 실행을 뜻하지는 않는다.**

③ O: 불균일 기억장치 액세스(NUMA) 구조는 여러 프로세서가 공유 주소 공간을 사용하지만, **접근하는 메모리의 위치에 따라 접근 시간이 달라지는** 구조이다. 각 프로세서가 독립적으로 명령어와 데이터를 처리할 수 있으므로 MIMD에 속한다.

④ X: 배열 프로세서는 하나의 제어 장치가 여러 처리 요소에 같은 명령을 전달하고, 각 처리 요소가 서로 다른 데이터에 그 연산을 수행하는 구조이다. 따라서 MIMD가 아니라 **SIMD**에 속한다. 여러 데이터를 동시에 처리한다는 점은 MIMD와 같지만, 명령어 흐름이 하나라는 점이 다르다.

NoteBase

문 5. 컴퓨터의 구성요소에 대한 설명으로 옳은 것만을 모두 고르면?

ㄱ. 입출력장치는 기계적 동작을 수반하기 때문에 동작 속도가 주기억장치보다 빠르다.
ㄴ. 중앙처리장치는 명령어 실행단계에서 제어장치, 내부 레지스터, 연산기를 필요로 한다.
ㄷ. 중앙처리장치는 명령어 인출단계에서 인출된 명령어를 저장하기 위한 명령어 레지스터와 다음에 실행할 명령어가 있는 기억장치의 주소를 저장할 프로그램 카운터를 필요로 한다.
ㄹ. 입출력장치는 중앙처리장치와 직접 데이터를 교환할 수 있으며, 데이터 교환은 반드시 중앙처리장치의 입출력 동작 제어에 의해서만 가능하다.

① ㄱ, ㄴ
② ㄱ, ㄹ
③ ㄴ, ㄷ
④ ㄷ, ㄹ

정답 3번

간단한 해설

명령어 레지스터는 **인출한 명령어**를, 프로그램 카운터는 **다음에 실행할 명령어의 주소**를 보관한다. 입출력에는 CPU가 매 데이터 전송을 직접 제어하지 않는 DMA 방식도 있다.

NoteBase

보기별 해설

ㄱ. X: 기계적 동작은 전자회로의 동작보다 느리므로, 기계적 동작을 수반하는 입출력장치는 일반적으로 주기억장치보다 **동작 속도가 느리다**. 예를 들어 하드디스크는 데이터를 읽을 위치로 헤드를 이동하고 원판이 회전하기를 기다리는 시간이 필요하지만, 주기억장치는 전자적으로 데이터를 읽고 쓴다.

ㄴ. O: 명령어 실행단계에서는 제어장치가 필요한 제어신호를 보내 각 구성요소의 동작을 지시하고, 내부 레지스터가 피연산자와 중간 결과 등을 임시로 보관하며, 연산기가 산술·논리 연산을 수행한다. 예를 들어 덧셈 명령을 실행할 때는 레지스터의 값을 연산기로 보내 더하고 그 결과를 저장하도록 제어한다.

ㄷ. O: 명령어 인출단계에서는 프로그램 카운터(PC)가 가리키는 주소의 명령어를 주기억장치에서 읽어 명령어 레지스터(IR)에 저장한다. **IR은 명령어 자체, PC는 명령어의 주소**를 보관한다는 차이가 있다. 순차 실행에서는 PC를 갱신하여 다음 명령어를 인출할 주소를 준비한다.

ㄹ. X: 입출력장치는 일반적으로 입출력 인터페이스나 제어기를 통해 CPU 또는 주기억장치와 데이터를 교환한다. 데이터 전송이 **반드시 CPU의 직접 제어로만 이루어지는 것은 아니다**. DMA(직접 기억장치 접근) 방식에서는 CPU가 전송에 필요한 주소·크기 등의 정보를 설정한 뒤, DMA 제어기가 입출력장치와 주기억장치 사이의 데이터 전송을 담당한다. 따라서 CPU가 데이터 하나하나의 전송을 직접 수행할 필요가 없다.

옳은 보기는 ㄴ과 ㄷ이며, 이 조합은 ③에 해당한다.

NoteBase

문 6. 유닉스 시스템 신호에 대한 설명으로 옳은 것은?

① SIGKILL: abort()에서 발생되는 종료 시그널

② SIGTERM: 잘못된 하드웨어 명령어를 수행하는 시그널

③ SIGILL: 터미널에서 CTRL + Z 할 때 발생하는 중지 시그널

④ SIGCHLD: 프로세스의 종료 혹은 중지를 부모에게 알리는 시그널

정답 4번

간단한 해설

SIGCHLD는 **자식 프로세스의 종료나 중지**를 부모 프로세스에 알리는 신호이다. 자식의 상태 변화를 통지하는 신호이므로, 자식을 종료하거나 중지시키는 명령과 구별한다.

선지별 해설

① X: abort()가 호출되면 발생하는 신호는 **SIGABRT**이다. SIGKILL은 프로세스를 강제로 종료하는 신호이며, 프로세스가 처리기를 등록하여 받거나 무시할 수 없다.

② X: 잘못된 명령어를 실행할 때 발생하는 신호는 **SIGILL**이다. SIGTERM은 프로세스에 종료를 요청하는 신호이며, 기본 동작은 종료이다. SIGKILL과 달리 프로세스가 신호 처리기를 통해 처리하거나 무시할 수 있다.

③ X: 일반적인 터미널 설정에서 CTRL + Z를 누르면 전경 작업에 **SIGTSTP**가 전달되며, 기본 동작은 실행을 일시 중지하는 것이다. SIGILL은 잘못된 명령어 실행을 알리는 신호로, 터미널의 중지 요청과 구별한다.

④ O: SIGCHLD는 **자식 프로세스의 상태 변화를 부모에게 통지**하는 신호이다. 자식 프로세스가 종료되거나 중지되면 부모가 이 신호를 통해 그 변화를 알 수 있다. 다만 부모가 SA_NOCLDSTOP 플래그를 설정한 경우 자식의 중지에 대한 통지는 억제된다.

NoteBase

문 7. 다음 설명에 해당하는 페이지 테이블 기술은?

물리 메모리의 프레임당 단 한 개의 페이지 테이블 항목을 할당함으로써 페이지 테이블이 차지하는 공간을 줄이는 기술

① 변환 참조 버퍼
② 계층적 페이지 테이블
③ 역 페이지 테이블
④ 해시 페이지 테이블

정답 3번

간단한 해설

물리 메모리의 프레임마다 항목을 하나씩 두는 방식이 역 페이지 테이블이다. 프로세스의 가상 페이지 수가 아니라 물리 프레임 수를 기준으로 테이블을 구성하여 공간 부담을 줄인다.

공통 해설

페이지는 가상 주소 공간을 일정한 크기로 나눈 단위이고, 프레임은 물리 메모리를 같은 크기로 나눈 단위이다. 일반적인 페이지 테이블이 가상 페이지를 기준으로 대응하는 물리 프레임을 기록하는 데 비해, 역 페이지 테이블은 **물리 프레임을 기준으로** 그 프레임에 저장된 페이지의 정보를 기록한다. 각 항목에는 해당 페이지의 가상 페이지 번호와 소속 프로세스를 식별하는 정보 등이 들어간다.

이 구조에서는 물리 프레임마다 항목이 하나씩 있으므로 테이블의 항목 수가 물리 프레임 수에 대응한다. 프로세스마다 가상 페이지 수에 맞춰 테이블을 두는 방식보다 페이지 테이블이 차지하는 공간을 줄일 수 있다. 따라서 제시문의 '물리 메모리의 프레임당 단 한 개의 페이지 테이블 항목'이라는 조건은 역 페이지 테이블을 가리킨다.

NoteBase

문 8. 다음 C 프로그램의 출력 결과는?

```
#include <stdio.h>
void main() {
      int x = 0x15213F10 >> 4;
      char y = (char) x;
      unsigned char z = (unsigned char) x;
      printf("%d, %u", y, z);
}
```

① −15, 15
② −241, 15
③ −15, 241
④ −241, 241

정답 3번

간단한 해설

오른쪽으로 4비트 이동한 x의 하위 8비트는 0xF1, 즉 1111 0001이다. 이를 **8비트 부호 있는 2의 보수**로 해석하는 y는 −15이고, **부호 없이** 해석하는 z는 241이다.

NoteBase

계산 과정

이 문항에서는 char가 8비트 부호 있는 정수이고, 형 변환 시 하위 8비트를 남겨 2의 보수로 해석하는 환경을 전제로 계산한다. C에서 char의 부호 여부와 범위를 벗어난 부호 있는 정수 변환 결과는 구현에 따라 달라질 수 있다.

x 계산: 16진수 한 자리는 4비트이므로, 양수 0x15213F10을 오른쪽으로 4비트 이동하면 맨 오른쪽의 16진수 한 자리가 버려져 x는 0x015213F1이 된다. 이 값의 **하위 8비트는 마지막 두 자리인 0xF1**이며, 2진수로는 1111 0001이다.

y 계산: (char)x로 변환하면 위 전제에서 하위 8비트 1111 0001이 남는다. 부호 있는 8비트 2의 보수에서는 **최상위 비트의 자릿값이 −128**이므로, y는 −128 + 64 + 32 + 16 + 1 = −15이다.

z 계산: (unsigned char)x에서도 하위 8비트 1111 0001이 남지만, unsigned char는 **모든 비트를 음이 아닌 값의 자릿수**로 해석한다. 따라서 z는 128 + 64 + 32 + 16 + 1 = 241이다.

printf는 y를 %d로, z를 %u로 출력하므로 출력 결과는 '−15, 241'이며, ③의 값과 일치한다.

NoteBase

문 9. 인터넷 계층에서 동작하는 프로토콜로서 오류보고, 상황보고, 경로 제어정보 전달 기능이 있는 프로토콜은?

① ICMP

② RARP

③ ARP

④ IGMP

정답 1번

간단한 해설

ICMP는 IP 통신을 보조하여 **오류·상태·경로 변경 정보를 제어 메시지로 전달**한다.

핵심 풀이

ICMP(Internet Control Message Protocol)는 인터넷 계층에서 IP 패킷 처리 중 발생한 오류와 통신 상태를 알리는 프로토콜이다. 목적지에 도달할 수 없을 때는 도달 불가 메시지로 오류를 보고하고, Echo 요청과 응답으로 상대 호스트의 응답 여부를 확인한다. 또한 Redirect 메시지는 호스트에 더 적절한 게이트웨이를 알려준다. 따라서 문제의 **오류보고·상황보고·경로 제어정보 전달**은 ICMP의 기능에 해당한다.

ARP와 RARP는 **IP 주소와 MAC 주소의 대응을 알아내는** 프로토콜이다. ARP는 IP 주소에 해당하는 MAC 주소를 찾고, RARP는 자신의 MAC 주소를 알고 있는 호스트가 자신의 IP 주소를 알아내는 데 사용한다. 오류나 통신 상태를 보고하는 기능과는 구별된다.

IGMP는 호스트와 인접한 멀티캐스트 라우터 사이에서 **멀티캐스트 그룹 가입 정보**를 전달한다. 여러 수신자에게 보내는 통신에서 어떤 그룹의 수신자가 있는지를 관리하므로, IP 패킷의 오류 보고를 담당하는 ICMP와 구별된다.

NoteBase

문 10. CPU의 제어장치에 해당하지 않는 것은?

① 순서 제어 논리 장치

② 명령어 해독기

③ 시프트 레지스터

④ 서브루틴 레지스터

정답 3번

간단한 해설

시프트 레지스터는 저장된 데이터의 **비트를 왼쪽이나 오른쪽으로 이동**시키는 장치로, 연산장치의 데이터 처리에 사용된다. 명령어를 해독하거나 실행 순서를 결정하는 제어장치의 구성 요소로 분류하지 않는다.

선지별 해설

① O: 순서 제어 논리 장치는 명령어 실행에 필요한 동작이 **정해진 순서와 시점**에 이루어지도록 제어한다. 예를 들어 명령어를 가져온 뒤 해독하고 실행하도록 각 단계의 제어 신호를 발생시키므로 제어장치에 해당한다.

② O: 명령어 해독기는 명령어의 **연산 코드**를 해석하여 어떤 연산을 수행해야 하는지 식별한다. 이를 바탕으로 해당 명령어 실행에 필요한 제어 동작이 선택되므로 제어장치에 해당한다.

③ X: 시프트 레지스터는 저장된 데이터의 비트를 왼쪽이나 오른쪽으로 이동시킨다. 이는 명령어의 실행 순서를 지시하는 기능이 아니라 **데이터 자체를 변환하는 기능**이므로, 이 문항의 기능별 구분에서는 제어장치가 아니라 연산장치의 구성 요소로 분류한다.

④ O: 서브루틴 레지스터는 마이크로프로그램 방식의 제어장치에서 마이크로서브루틴을 호출할 때 **복귀할 제어기억장치 주소**를 저장한다. 서브루틴 실행이 끝나면 이 주소를 이용하여 호출한 지점 다음의 마이크로명령으로 돌아가므로 제어 흐름을 관리하는 구성 요소이다.

NoteBase

문 11. 시간적으로 연속적인 아날로그 신호에 대해 일정한 시간 간격으로 아날로그 신호 값을 추출하는 과정은?

① 표본화
② 양자화
③ 부호화
④ 자동화

정답 1번

간단한 해설

아날로그 신호에서 **일정한 시간 간격으로 신호 값을 추출**하는 과정이 표본화이다. 시간축을 일정 간격으로 나누어 각 시점의 값을 얻는다는 점이 핵심이다.

개념 구분

표본화: 시간적으로 연속인 아날로그 신호에서 **일정한 시간 간격마다 순간적인 신호 값**을 추출하는 과정이다. 예를 들어 1 ms마다 신호 값을 읽으면 연속된 파형에서 0 ms, 1 ms, 2 ms 등의 값만 얻는다. 이때 시간을 이산적으로 나누었을 뿐, 추출한 값의 크기를 정해진 단계로 바꾼 것은 아니다.

양자화: 표본화로 얻은 신호 값의 크기를 **미리 정한 유한한 단계의 값**으로 대응시키는 과정이다. 따라서 '언제 값을 뽑는가'에 관한 표본화와 '값의 크기를 어느 단계로 나타내는가'에 관한 양자화를 구별한다.

부호화: 양자화된 각 값을 **이진 부호로 표현**하는 과정이다. 문제는 값의 크기를 단계로 나누거나 이진 부호로 바꾸는 것이 아니라 일정한 시간 간격으로 값을 추출하는 과정을 묻고 있으므로 표본화에 해당한다.

NoteBase

문 12. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
int funa(int);
void main() {
     printf("%d, %d", funa(5), funa(6));
     return 0;
}

int funa(int n) {
     if(n > 1)
          return (n + (funa(n-2)));
     else
          return (n % 2);
}
```

① 5, 6

② 9, 12

③ 15, 21

④ 120, 720

정답 2번

간단한 해설

funa는 인수를 **2씩 줄여 재귀 호출하고 현재 n을 더한다.** 따라서 funa(5) = 9, funa(6) = 12가 되어 '9, 12'가 출력된다.

NoteBase

계산 과정

재귀 계산 규칙: n은 각 호출에서 전달받은 정수이다. n > 1이면 funa(n) = n + funa(n − 2)를 계산한다. 즉, **인수를 2 줄인 호출의 반환값에 현재 n을 더한다**. n이 1 또는 0에 도달하면 재귀 호출을 멈추고 n % 2를 반환한다. %는 나머지 연산자이므로 funa(1) = 1, funa(0) = 0이다.

funa(5) 계산: 인수는 5 → 3 → 1로 줄어든다. 마지막 호출이 1을 반환하면, 그 값을 기다리던 호출부터 차례로 계산한다. funa(3) = 3 + 1 = 4이고, 이 4는 인수 3에서 시작한 호출의 반환값이다. 따라서 funa(5) = 5 + 4 = 9이다.

funa(6) 계산: 인수는 6 → 4 → 2 → 0으로 줄어든다. 마지막 호출이 0을 반환하므로 funa(2) = 2 + 0 = 2, funa(4) = 4 + 2 = 6이 된다. 이 6을 받아 funa(6) = 6 + 6 = 12를 반환한다.

출력: printf의 첫 번째 %d에는 funa(5)의 반환값 9가, 두 번째 %d에는 funa(6)의 반환값 12가 들어간다. 두 값 사이에는 형식 문자열의 쉼표와 공백이 출력되므로 실행 결과는 '9, 12'이며 ②와 일치한다.

NoteBase

문 13. 다음에서 설명하는 해시 함수는?

탐색키 값을 여러 부분으로 나눈 후 각 부분의 값을 더하거나 XOR(배타적 논리합) 연산하여 그 결과로 주소를 취하는 방법

① 숫자분석함수
② 제산함수
③ 중간제곱함수
④ 폴딩함수

정답 4번

간단한 해설

탐색키를 여러 부분으로 나누고 그 값들을 **덧셈이나 XOR로 결합**하여 주소를 만드는 방식이 폴딩함수이다. 키의 각 부분을 하나의 값으로 접어 합친다는 점이 핵심이다.

공통 해설

폴딩함수는 탐색키를 여러 부분으로 나눈 뒤, **각 부분의 값을 더하거나 XOR로 결합**하여 해시 주소를 구하는 방법이다. 키 전체를 한 덩어리로 처리하는 대신, 나뉜 부분들의 정보를 하나의 값에 모은다. 지문의 '여러 부분으로 나눈 후'와 '더하거나 XOR 연산'은 이 분할·결합 과정을 함께 나타내므로 폴딩함수의 정의에 해당한다.

예를 들어 탐색키 123456을 12, 34, 56으로 나누어 더하면 결합값은 12 + 34 + 56 = 102가 된다. XOR를 사용하는 경우에는 나눈 값들을 이진수로 나타내어 같은 자리의 비트가 서로 다르면 1, 같으면 0이 되도록 결합한다. 덧셈과 XOR는 결합 연산이 다르지만, 키를 나눈 부분들을 합쳐 하나의 값을 만든다는 공통 구조를 가진다.

NoteBase

문 14. (가)~(다)에 해당하는 말을 바르게 연결한 것은?

(가) 컴퓨터가 데이터를 통해 스스로 학습하여 예측이나 판단을 제공하는 기술
(나) 인간의 지적 능력을 컴퓨터를 통해 구현하는 기술
(다) 인공 신경망을 활용하는 개념으로, 여러 계층의 신경망을 구성해 학습을 효과적으로 수행하는 기술

① (가): 인공지능, (나): 머신러닝, (다): 딥러닝
② (가): 인공지능, (나): 딥러닝, (다): 머신러닝
③ (가): 머신러닝, (나): 인공지능, (다): 딥러닝
④ (가): 머신러닝, (나): 딥러닝, (다): 인공지능

정답 3번

간단한 해설

인공지능은 인간의 지적 능력을 구현하는 넓은 개념이며, 그중 **데이터에서 학습하는 방식**이 머신러닝이다.
딥러닝은 머신러닝 중 **여러 계층의 인공 신경망**을 사용하는 방식이다.

대응 관계 해설

(가) 머신러닝: 데이터를 통해 규칙이나 패턴을 학습하고, 이를 새로운 데이터의 예측이나 판단에 이용하는 기술이다. 핵심 단서는 **데이터를 통한 학습**이다. 예를 들어 스팸 메일과 정상 메일의 사례를 학습한 뒤 새 메일이 스팸인지 판단하는 방식이 이에 해당한다.

(나) 인공지능: 추론·학습·문제 해결 등 **인간의 지적 능력을 컴퓨터로 구현**하는 기술을 넓게 가리킨다. 데이터를 학습하는 특정 방법에 한정되지 않으며, 머신러닝과 딥러닝을 포함하는 상위 개념이다.

(다) 딥러닝: 머신러닝의 한 분야로, **여러 계층의 인공 신경망**을 이용해 데이터의 특징과 패턴을 학습한다. 따라서 단순히 데이터를 학습한다는 설명보다 신경망의 여러 계층을 구성한다는 조건이 딥러닝을 구별하는 단서이다.

따라서 (가) 머신러닝, (나) 인공지능, (다) 딥러닝의 대응은 ③과 일치한다.

NoteBase

문 15. 구조적 개발 방법론에서 사용자 요구사항을 분석한 후 결과를 표현할 때 사용되는 도구에 대한 설명으로 옳은 것은?

① 자료흐름도에서 자료저장소는 원으로 표현한다.

② 자료사전은 계획(ISP), 분석(BAA), 설계(BSD), 구축(SC)의 절차로 작성한다.

③ 자료사전에서 사용하는 기호 중 ( )는 선택에 사용되는 기호이다.

④ 소단위 명세서를 작성하는 도구에는 구조적언어, 의사결정표 등이 있다.

정답 4번

간단한 해설

소단위 명세서는 자료흐름도에서 더 이상 분해하지 않는 **최하위 처리의 내부 논리**를 구체적으로 기술한다. 처리 절차는 구조적언어로, 조건의 조합에 따른 수행 동작은 의사결정표로 표현할 수 있다.

선지별 해설

① X: 자료흐름도에서 원은 자료를 입력받아 변환하는 **처리 과정**을 나타낸다. 자료를 보관하는 자료저장소는 원이 아니라 두 개의 평행선 등으로 표현한다.

② X: 정보전략계획(ISP) → 업무영역분석(BAA) → 업무시스템설계(BSD) → 시스템구축(SC)은 **정보공학 방법론의 개발 단계**이다. 자료사전은 이러한 단계의 이름이 아니라 자료흐름도에 나타나는 자료의 명칭·의미·구성 등을 일관되게 정의한 것이다.

③ X: 자료사전에서 ( )는 해당 자료 항목을 **생략할 수 있음**을 나타낸다. 여러 자료 항목 중 하나를 선택하는 표기는 [ ]이다. 따라서 항목이 없어도 되는 생략과 여러 대안 중 하나를 고르는 선택을 구분해야 한다.

④ O: 소단위 명세서는 자료흐름도의 최하위 처리가 입력 자료를 어떤 논리로 처리하여 출력 자료를 만드는지 기술한다. **구조적언어**는 순차·선택·반복 구조로 처리 절차를 서술하고, **의사결정표**는 조건의 조합과 그에 따라 수행할 동작을 대응시킨다. 따라서 두 도구 모두 소단위 명세서 작성에 사용할 수 있다.

NoteBase

문 16. 다음 내용에 해당하는 법칙은?

주식회사의 주가를 보면 일일 가격은 급격히 변동할 수 있다. 하지만 긴 기간의 움직임을 보면 상승, 하락 또는 변동 없는 추세를 보인다.

① 자기 통제의 법칙
② 복잡도 증가의 법칙
③ 피드백 시스템의 법칙
④ 지속적 변경의 법칙

정답 1번

간단한 해설

일일 가격이 급격히 변해도 긴 기간에는 일정한 추세가 드러난다는 점이 **자기 통제의 법칙**에 해당한다. 핵심은 단기 변동 속에서도 전체적인 변화 경향을 파악할 수 있다는 것이다.

공통 해설

레만의 소프트웨어 진화 법칙에서 자기 통제의 법칙은 진화 과정이 단기적으로 불규칙해 보여도, 전체적으로는 **통계적으로 파악할 수 있는 경향과 규칙성**을 나타낸다는 것이다. 여기서 자기 통제는 변화가 멈추거나 값이 언제나 일정하게 유지된다는 뜻이 아니다.

제시문의 주가는 하루 단위로는 급격히 변하지만, 긴 기간으로 보면 상승·하락·변동 없음이라는 추세를 보인다. 이 사례는 개별 변동과 장기적인 경향이 함께 나타날 수 있음을 보여 주므로 자기 통제의 법칙에 대응한다. 상승과 하락도 포함하므로, 장기 추세가 있다는 말을 가격이 항상 같은 수준을 유지한다는 뜻으로 해석해서는 안 된다.

NoteBase

문 17. 그림과 같이 S 테이블과 T 테이블이 있을 때, SQL 실행 결과는?

S

a	b
1	가
2	나
3	다

T

c	d
나	X
다	Y
라	Z

```
SELECT S.a, S.b, T.d
FROM S
LEFT JOIN T
ON S.b = T.c
```

①

a	b	d
1	가	(NULL)
2	나	X
3	다	Y

②

a	b	d
2	나	X
3	다	Y
1	가	(NULL)

③

a	b	d
1	가	(NULL)
2	나	X
3	다	Y
(NULL)	라	Z

NoteBase

④

a	b	d
2	나	X
3	다	Y
(NULL)	라	Z

정답 1, 2번

간단한 해설

LEFT JOIN은 **왼쪽 S의 모든 행을 보존**하므로, T에 대응하는 '가'가 없어도 (1, 가, NULL)이 남는다. 결과는 (1, 가, NULL), (2, 나, X), (3, 다, Y)의 세 행이며, **ORDER BY가 없어 행의 출력 순서는 보장되지 않는다**.

계산 과정

조인 기준: FROM S LEFT JOIN T에서 S가 왼쪽 테이블이다. ON S.b = T.c는 S의 b와 T의 c가 같은 행끼리 연결하라는 조건이다. **일치하는 T의 행이 없어도 S의 행은 남기고**, 그 행에 대응하는 T의 열은 NULL로 채운다. SELECT S.a, S.b, T.d에 따라 연결 결과에서 a, b, d 세 열만 출력한다.

S의 (1, 가): T의 c에는 '나', '다', '라'만 있어 '가'와 일치하는 행이 없다. S의 a와 b는 유지하고 T.d를 NULL로 채우므로 출력 행은 (1, 가, NULL)이다.

S의 (2, 나): T의 (나, X)와 연결된다. S.a는 2, S.b는 '나', T.d는 X이므로 출력 행은 (2, 나, X)이다.

S의 (3, 다): T의 (다, Y)와 연결된다. S.a는 3, S.b는 '다', T.d는 Y이므로 출력 행은 (3, 다, Y)이다.

결과와 선택지 연결: 각 S 행에서 한 행씩 나와 총 3행이다. T의 (라, Z)는 대응하는 S의 행이 없으며, LEFT JOIN은 **오른쪽 T에만 있는 행을 추가하지 않는다**. 따라서 Z를 포함한 행은 결과에 없다. ①과 ②는 모두 (1, 가, NULL), (2, 나, X), (3, 다, Y)의 세 행으로 구성되어 있고 배치 순서만 다르다. **ORDER BY가 없으므로 특정 행 순서가 보장되지 않아** 두 선택지 모두 가능한 실행 결과이다.

NoteBase

문 18. 운영체제 시스템 호출에 대한 설명으로 옳지 않은 것은?

① fork( )는 실행 중인 프로세스를 복사하는 함수이다.

② fork( ) 호출 시 부모 프로세스와 자식 프로세스가 차지하는 메모리 위치는 동일하다.

③ exec( )는 이미 만들어진 프로세스의 구조를 재활용하는 함수이다.

④ exec( ) 호출에 사용되는 함수 중 wait( )는 프로세스 종료 대기를 처리한다.

정답 2번

간단한 해설

fork()는 부모의 메모리 내용을 복사하지만, 부모와 자식은 **독립된 주소 공간**을 갖는다. 복사된 내용이 같다는 것이 두 프로세스가 같은 메모리 공간을 사용한다는 뜻은 아니다.

선지별 해설

① O: fork()는 실행 중인 부모 프로세스를 복제하여 **새로운 자식 프로세스**를 만든다. 자식은 부모의 메모리 내용과 여러 실행 환경을 물려받지만, 부모와 다른 프로세스 식별자(PID)를 갖는다.

② X: 부모와 자식은 복제 직후 메모리 내용이 같더라도 **서로 독립된 주소 공간**에서 실행된다. 따라서 '차지하는 메모리 위치는 동일하다'를 '메모리 내용은 복사되지만 주소 공간은 독립적이다'로 고쳐야 한다. 리눅스의 쓰기 시 복사(copy-on-write) 구현에서는 물리 페이지를 일시적으로 공유할 수 있지만, 이것도 두 프로세스가 하나의 주소 공간을 사용한다는 뜻은 아니다.

③ O: exec()는 새 프로세스를 만드는 대신, **기존 프로세스의 프로그램 이미지를 다른 프로그램으로 교체**한다. 프로그램 이미지는 실행할 코드와 데이터 등의 메모리 내용을 뜻한다. 호출 전의 PID는 유지되므로 '프로세스의 구조를 재활용한다'고 표현할 수 있으며, 기존 프로그램의 코드를 그대로 실행한다는 뜻은 아니다.

④ O: wait()는 부모 프로세스가 **자식 프로세스의 종료를 기다리고 종료 상태를 회수**하는 함수이다. fork()로 생성한 자식이 exec()로 다른 프로그램을 실행하는 경우에도 부모는 wait()로 그 자식의 종료를 기다릴 수 있다. 다만 wait() 자체가 exec 계열 함수인 것은 아니며, 별도의 프로세스 대기 함수이다.

NoteBase

문 19. SQL 뷰에 대한 설명으로 옳은 것은?

① 복잡한 질의를 간단하게 표현할 수 있게 한다.

② 데이터 무결성을 보장하지만 독립성을 제공하지는 않는다.

③ 제거할 때는 DELETE문을 사용한다.

④ 동일한 데이터에 대해 하나의 뷰만 생성 가능하다.

정답 1번

간단한 해설

뷰에 복잡한 조인이나 집계 질의를 미리 정의해 두면, 사용자는 **뷰 이름을 이용한 간단한 질의**로 그 결과를 조회할 수 있다.

선지별 해설

① O: 뷰는 하나 이상의 테이블을 조회하는 질의로 정의되는 가상 테이블이다. 조인이나 집계가 포함된 복잡한 질의를 뷰로 정의하면, 사용자는 **복잡한 질의 내용을 매번 작성하지 않고** 뷰를 조회할 수 있다. 예를 들어 부서별 급여 합계를 구하는 질의를 뷰로 만들어 두면, 이후에는 그 뷰에서 부서명과 합계만 간단히 조회할 수 있다.

② X: 뷰는 **논리적 데이터 독립성을 지원한다**. 기본 테이블의 구조가 변경되더라도 뷰 정의를 적절히 조정하여 사용자에게 제공되는 구조를 유지하면, 뷰를 사용하는 응용 프로그램의 변경을 줄일 수 있다. 따라서 '독립성을 제공하지는 않는다'는 설명이 잘못되었다.

③ X: 뷰 자체를 제거할 때는 **DROP VIEW문**을 사용한다. DELETE문은 데이터의 행을 삭제하는 명령이므로, 뷰 정의 자체를 제거하는 명령과 구별해야 한다.

④ X: **동일한 데이터에 여러 개의 뷰**를 정의할 수 있다. 사용자나 업무 목적에 따라 같은 기본 테이블에서 서로 다른 열이나 조건에 맞는 행을 보여 주도록 각각의 뷰를 만들 수 있다.

NoteBase

문 20. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
int C(int v) {
     printf("%d ", v);
     return 1;
}

int main() {
     int a = -2;
     int b = !a;
     printf("%d %d %d %d ", a, b, a&&b, a||b);
     if(b && C(10))
           printf("A ");
     if(b & C(20))
           printf("B ");
     return 0;
}
```

① -2 0 0 1 20

② -2 0 0 1 10 20

③ -2 1 0 1 10 20

④ -2 2 1 1 10 A 20 B

정답 1번

간단한 해설

a = -2는 참으로 취급되므로 b = !a는 0이며, 첫 출력은 -2 0 0 1이다. **&&의 단락 평가**로 C(10)은 실행되지 않지만, **비트 AND인 &**는 C(20)을 실행하므로 뒤에 20만 추가된다.

NoteBase

계산 과정

함수 C(v)는 전달받은 v를 출력한 뒤 1을 반환한다. 따라서 **함수 안에서 출력하는 값과 반환하는 값**을 구분해야 한다. C(20)이 호출되면 20을 출력하지만, 호출식의 값은 1이다.

초기값: C 언어의 조건 판단에서는 0은 거짓, **0이 아닌 값은 참**으로 취급한다. a는 -2이므로 참이고, 논리 부정 연산 !a의 결과는 0이다. 따라서 a = -2, b = 0이다.

첫 printf: 논리 AND 연산 a&&b는 두 값이 모두 참일 때만 1이므로 0이다. 논리 OR 연산 a||b는 하나라도 참이면 1이므로 1이다. 논리 연산의 결과는 피연산자 -2 자체가 아니라 **0 또는 1**이다. 이에 따라 a, b, a&&b, a||b가 차례로 -2 0 0 1로 출력된다.

첫 번째 if: b && C(10)에서 왼쪽 b가 0이므로 전체 조건은 이미 거짓이다. &&는 **왼쪽이 거짓이면 오른쪽을 평가하지 않는 단락 평가**를 하므로 C(10)은 호출되지 않아 10이 출력되지 않는다. 조건이 거짓이므로 A도 출력되지 않는다.

두 번째 if: b & C(20)의 &는 비트 AND 연산자이므로 **단락 평가 없이 양쪽 피연산자를 모두 평가**한다. 따라서 C(20)이 호출되어 20을 출력하고 1을 반환한다. 이후 조건식의 값은 0 & 1 = 0이 되므로 B는 출력되지 않는다.

전체 출력은 -2 0 0 1 20이며, ①과 일치한다.

NoteBase