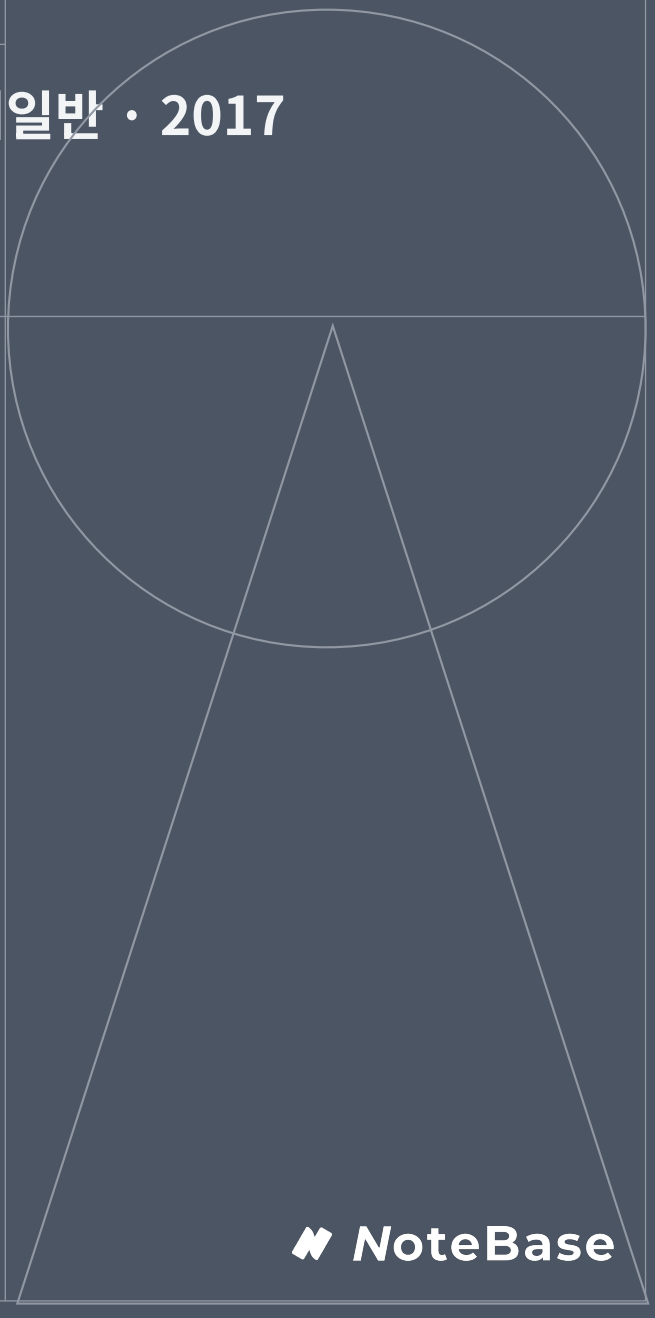
기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2017



NoteBase

기출OX 해설집

공무원/지방직/9급 · 컴퓨터일반 · 2017

공무원/지방직/9급 · 컴퓨터일반 · 2017

문 1. 네트워크 프로토콜에 대한 설명으로 옳지 않은 것은?

① TCP와 UDP는 전송 계층에 속하는 프로토콜로서 데이터 전송의 신뢰성을 보장한다.

② IP는 네트워크 호스트의 주소 지정과 경로 설정을 담당하는 네트워크 계층 프로토콜이다.

③ SMTP는 전자메일 전송을 위한 응용 계층 프로토콜이다.

④ IPv4에서 예상되는 IP 주소의 고갈 문제 해결을 주요 목적으로 IPv6가 제안되었다.

정답 1번

간단한 해설

TCP와 UDP는 모두 전송 계층 프로토콜이지만, **UDP는 데이터 전송의 신뢰성을 보장하지 않는다**. TCP는 확인 응답과 재전송 등을 통해 신뢰성 있는 전송을 제공한다.

선지별 해설

① X: TCP와 UDP는 모두 전송 계층에 속하지만, **신뢰성 있는 전송을 제공하는 것은 TCP**이다. TCP는 순서 번호와 확인 응답을 이용해 데이터의 수신 여부와 순서를 관리하고, 손실된 데이터를 재전송한다. UDP는 이러한 전달 보장과 순서 보장 기능을 자체적으로 제공하지 않으므로, 두 프로토콜 모두 신뢰성을 보장한다는 설명은 잘못이다.

② O: IP는 네트워크 계층 프로토콜로, **IP 주소를 이용한 논리적 주소 지정**과 네트워크 간 패킷 전달을 담당한다. 라우터는 목적지 IP 주소와 라우팅 정보를 바탕으로 패킷을 보낼 다음 경로를 결정한다.

③ O: SMTP는 **전자메일을 보내고 메일 서버 사이에서 전달**하기 위한 응용 계층 프로토콜이다. 메일 클라이언트가 서버로 메일을 제출하거나 메일 서버가 다른 메일 서버로 메일을 전달할 때 사용한다.

④ O: IPv4는 32비트 주소를 사용하므로 사용할 수 있는 주소 공간에 한계가 있다. IPv6는 주소 길이를 **128비트로 확장**하여 훨씬 넓은 주소 공간을 제공하며, IPv4 주소 고갈 문제의 해결이 주요 도입 목적이다.

NoteBase

문 2. 하드디스크에 대한 설명으로 옳지 않은 것은?

① 하드디스크는 데이터접근 방식이 직접접근 방식인 보조기억장치이다.

② 바이오스(BIOS)는 하드디스크에 저장된다.

③ 하드디스크는 주기억장치보다 접근 속도가 느리다.

④ 하드디스크는 전원이 꺼져도 저장된 데이터가 지워지지 않는다.

정답 2번

간단한 해설

바이오스(BIOS)는 하드디스크가 아니라 **메인보드의 ROM이나 플래시 메모리**에 저장된다. 전원을 켜면 하드웨어를 초기화하고 운영체제의 부팅을 준비하는 펌웨어이다.

선지별 해설

① O: 하드디스크는 프로그램과 데이터를 보관하는 보조기억장치로, **원하는 데이터가 있는 위치를 지정하여 접근**할 수 있다. 자기테이프처럼 앞부분부터 순서대로 읽어야 하는 순차접근 방식과 구별된다.

② X: 바이오스(BIOS)는 전원을 켰을 때 하드웨어를 점검·초기화하고 운영체제의 부팅을 준비하는 펌웨어이다. 하드디스크가 아니라 **메인보드의 ROM이나 플래시 메모리**에 저장된다.

③ O: 하드디스크는 데이터를 읽기 위해 헤드를 이동시키고 원하는 데이터가 있는 부분이 회전하여 도착하기를 기다려야 한다. 이러한 **기계적 동작에 따른 지연** 때문에 전자적으로 접근하는 주기억장치인 RAM보다 접근 속도가 느리다.

④ O: 하드디스크는 디스크 표면의 자성 상태로 데이터를 기록하는 **비휘발성 기억장치**이다. 데이터를 유지하는 데 계속 전원을 공급할 필요가 없으므로, 전원을 꺼도 저장된 데이터가 유지된다.

NoteBase

문 3. 가상 사설 네트워크(VPN : Virtual Private Network)에 대한 설명으로 옳지 않은 것은?

① 터널링(tunneling) 기술을 사용한다.

② 전용회선 기반 사설 네트워크보다 구축 및 유지 비용이 높다.

③ 암호화 기술을 사용한다.

④ VPN 기능은 방화벽이나 라우터에 내장될 수 있다.

정답 2번

간단한 해설

VPN은 인터넷 같은 공중망을 이용하므로 전용회선을 별도로 사용하는 사설 네트워크보다 **일반적으로 구축 및 유지 비용이 낮다**. 따라서 비용이 '높다'는 설명은 비용 절감이라는 VPN의 장점을 거꾸로 서술한 것이다.

선지별 해설

① O: VPN은 **터널링**을 이용해 공중망 위에 논리적인 통신 경로를 만든다. 터널링은 원래의 패킷을 다른 패킷 안에 넣어 전송하는 캡슐화 방식으로, 수신 측에서 바깥쪽 정보를 제거하여 원래 패킷을 복원한다.

② X: VPN은 인터넷 같은 기존 공중망을 이용하므로 지점 사이에 전용회선을 별도로 확보하는 비용을 줄일 수 있다. 따라서 전용회선 기반 사설 네트워크보다 구축 및 유지 비용이 **일반적으로 낮다**고 고쳐야 한다.

③ O: 공중망에서는 전송 중인 데이터가 노출될 수 있으므로 VPN은 **암호화로 통신 내용의 기밀성을 보호**한다. 터널링이 패킷을 감싸 전달하는 방식이라면, 암호화는 권한 없는 사람이 데이터 내용을 알아보기 어렵게 만드는 기술이다.

④ O: VPN 기능은 별도 전용 장비뿐 아니라 **방화벽이나 라우터에 통합**하여 제공할 수 있다. VPN 기능이 내장된 장비는 네트워크 경계에서 터널을 설정하고 암호화·복호화 등을 수행한다.

NoteBase

문 4. 다음은 폭포수 모델에서 제시하는 소프트웨어 개발 단계들 중 일부에 대한 설명이다. 제시된 소프트웨어 개발 단계를 순서대로 바르게 나열한 것은?

ㄱ. 시스템 구조, 프로그램, 인터페이스를 설계한다.
ㄴ. 소프트웨어를 이용하면서 문제점을 수정하거나 새로운 기능을 추가한다.
ㄷ. 요구대로 소프트웨어가 적합하게 작동하는지 확인한다.
ㄹ. 사용자의 요구사항을 파악한다.

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

정답 3번

간단한 해설

사용자의 **요구사항이 설계의 기준**이 되며, 설계에 따라 구현한 소프트웨어가 그 요구를 충족하는지 검사한다. 사용 중 발견한 문제를 수정하거나 기능을 추가하는 **유지보수는 운용 단계**에서 이루어진다.

올바른 순서

ㄹ → ㄱ → ㄷ → ㄴ

ㄹ. 요구사항 분석: 사용자가 소프트웨어에 원하는 기능과 조건을 파악한다. **무엇을 만들어야 하는지** 정해야 이를 실현할 구조와 동작을 설계할 수 있다.

ㄱ. 설계: 분석한 요구사항을 바탕으로 시스템 구조, 프로그램, 인터페이스를 구체화한다. 요구사항이 '무엇을 할 것인가'라면 설계는 **어떻게 구현할 것인가**를 정하는 단계이다.

ㄷ. 테스트: 설계에 따라 구현된 소프트웨어가 **요구사항대로 작동하는지** 확인한다. 검사할 소프트웨어가 먼저 구현되어 있어야 하므로 설계보다 뒤에 놓인다.

ㄴ. 유지보수: 소프트웨어를 실제로 이용하면서 발견한 문제점을 수정하거나 새로운 기능을 추가한다. **사용 중인 소프트웨어를 수정·개선**하는 활동이므로 개발 결과를 검사한 뒤의 운용 단계에 해당한다.

NoteBase

문 5. 입출력과 관련하여 폴링(polling) 방식과 인터럽트(interrupt) 방식에 대한 설명으로 옳지 않은 것은?

① 폴링 방식에서는 프로세서가 입출력을 위해 입출력장치의 상태를 반복적으로 검사한다.

② 인터럽트 방식은 폴링 방식 대비 프로세서의 시간을 낭비하는 단점이 있다.

③ 인터럽트 방식에서는 인터럽트 간에 우선순위를 둘 수 있다.

④ 인터럽트 방식에서는 인터럽트 처리를 위해 인터럽트 처리 루틴을 호출한다.

정답 2번

간단한 해설

인터럽트 방식은 입출력장치의 상태를 계속 확인하는 대신, 장치가 처리를 요청할 때 대응하므로 **반복 검사에 드는 프로세서 시간의 낭비를 줄인다**. '폴링 방식 대비 프로세서의 시간을 낭비한다'는 설명은 두 방식의 일반적인 특징을 반대로 서술한 것이다.

선지별 해설

① O: 폴링 방식에서는 프로세서가 입출력장치의 상태 정보를 **반복적으로 읽어** 데이터 전송 준비나 작업 완료 여부를 확인한다. 장치가 아직 준비되지 않았더라도 검사를 계속하면 그만큼 프로세서 시간이 소모된다.

② X: 인터럽트 방식에서는 입출력장치가 서비스가 필요한 시점에 인터럽트 신호를 보내므로, 프로세서가 장치 상태를 계속 검사할 필요가 없다. 프로세서는 그동안 다른 작업을 수행할 수 있어 **폴링의 반복 검사로 인한 시간 낭비를 줄인다**. 인터럽트 처리 자체에도 시간이 들지만, 이를 폴링보다 프로세서 시간을 낭비하는 방식이라고 설명하는 것은 부적절하다.

③ O: 여러 인터럽트 요청이 있을 때 어느 요청부터 처리할지 결정하도록 **인터럽트 간 우선순위**를 설정할 수 있다. 예를 들어 여러 장치가 동시에 처리를 요청하면 우선순위가 높은 요청을 먼저 처리하도록 구성할 수 있다.

④ O: 프로세서가 인터럽트를 받아들이면 실행 중이던 작업의 복귀에 필요한 상태를 보존하고, 해당 **인터럽트 처리 루틴**으로 제어를 옮겨 필요한 처리를 수행한다. 처리가 끝나면 보존한 상태를 복원하여 중단했던 작업을 재개한다.

NoteBase

문 6. 다음은 배열로 구현한 스택 자료구조의 push() 연산과 pop() 연산이다. ㉠과 ㉡에 들어갈 코드가 옳게 짝지어진 것은?

```
#define ARRAY_SIZE 10
#define IsFull() ((top == ARRAY_SIZE-1) ?  1: 0)
#define IsEmpty() ((top == -1) ? 1: 0)

int a[ARRAY_SIZE];
int top = -1;

void push(int d) {
        if( IsFull() )
                printf("STACK FULL\n");
        else

                ㉠

}

int pop() {
        if( IsEmpty() )
                printf("STACK EMPTY\n");
        else

                ㉡

}
```

① ㉠: a[++top] = d;, ㉡: return a[--top];

② ㉠: a[++top] = d;, ㉡: return a[top--];

③ ㉠: a[--top] = d;, ㉡: return a[++top];

④ ㉠: a[top--] = d;, ㉡: return a[top++];

정답 2번

간단한 해설

top은 현재 맨 위 자료의 인덱스이므로, push에서는 **top을 먼저 증가시킨 뒤 저장**해야 한다. pop에서는 **현재 top의 자료를 꺼내고 top을 감소**시켜야 하므로 후위 감소 연산자 top--를 사용한다.

NoteBase

빈칸별 해설

스택은 가장 나중에 넣은 자료를 먼저 꺼내는 후입선출 구조이다. 제시된 코드에서 배열 a의 인덱스는 0~9이고, top은 현재 맨 위 자료의 인덱스이다. top이 -1이면 비어 있고, 9이면 가득 차 있다. push와 pop은 각각 가득 찬 상태와 빈 상태를 검사한 뒤 else 부분에서 실제 삽입·삭제를 수행한다.

㉠: a[++top] = d;를 넣는다. 삽입할 때는 현재 맨 위 칸의 바로 다음 칸에 새 자료를 저장해야 하므로 **top을 1 증가시킨 값을 인덱스로 사용**한다. 전위 증가 연산자 ++top이 이 동작을 수행한다. 예를 들어 빈 스택에서 7을 넣으면 top이 -1에서 0으로 바뀌고 a[0]에 7이 저장된다. --top이나 top--를 사용하면 인덱스가 감소하는 방향으로 움직이므로 이 코드의 삽입 방식에 맞지 않으며, 빈 상태에서는 유효한 배열 범위를 벗어난 곳에 저장하려 하게 된다.

㉡: return a[top--];를 넣는다. 삭제할 때는 **감소 전 top의 자료를 반환**하고 top을 1 줄여야 한다. 후위 감소 연산자 top--는 배열 인덱스로 기존 값을 사용하면서 top을 감소시킨다. 앞의 예에서 pop을 실행하면 a[0]의 7을 반환하고 top은 -1이 되어 빈 상태로 돌아간다. a[--top]은 top을 먼저 줄여 맨 위 자료가 아닌 아래 칸을 읽으며, 자료가 하나뿐이면 a[-1]을 읽으려 한다. ++top이나 top++는 삭제하면서 top을 증가시키므로 남은 자료의 맨 위 위치를 올바르게 나타내지 못한다.

㉠의 a[++top] = d;와 ㉡의 return a[top--];를 짝지은 것은 ②이다.

NoteBase

문 7. 32비트 16진수 정수 $302AF567_{(16)}$이 메모리 주소 $200_{(16)}$부터 시작하는 4바이트에 저장되어 있다. 리틀 엔디안(little endian) 방식을 사용하는 시스템에서 메모리 주소와 그 주소에 저장된 8비트 데이터가 옳게 짝지어진 것은? (단, 바이트 단위로 주소가 지정된다)

① $200_{(16)}$: $67_{(16)}$, $201_{(16)}$: $F5_{(16)}$, $202_{(16)}$: $2A_{(16)}$, $203_{(16)}$: $30_{(16)}$
② $200_{(16)}$: $F5_{(16)}$, $201_{(16)}$: $67_{(16)}$, $202_{(16)}$: $30_{(16)}$, $203_{(16)}$: $2A_{(16)}$
③ $200_{(16)}$: $30_{(16)}$, $201_{(16)}$: $2A_{(16)}$, $202_{(16)}$: $F5_{(16)}$, $203_{(16)}$: $67_{(16)}$
④ $200_{(16)}$: $2A_{(16)}$, $201_{(16)}$: $30_{(16)}$, $202_{(16)}$: $67_{(16)}$, $203_{(16)}$: $F5_{(16)}$

정답 1번

간단한 해설

리틀 엔디안은 **가장 하위 바이트를 가장 낮은 주소에** 저장한다. 16진수 302AF567의 가장 하위 바이트는 67이므로, 시작 주소인 16진수 200에는 67이 저장된다.

대응 관계 해설

이하 주소와 데이터는 모두 16진수로 표시한다. 16진수 한 자리는 4비트이므로 8비트인 1바이트는 두 자리로 나타낸다. 32비트 정수 302AF567을 바이트로 나누면 상위부터 30, 2A, F5, 67이다. 리틀 엔디안은 **하위 바이트부터 주소가 증가하는 방향으로** 저장하며, 바이트 내부의 두 자리 순서는 바꾸지 않는다.

주소 200 → 데이터 67: 시작 주소에는 가장 하위인 첫 번째 바이트 67이 저장된다.

주소 201 → 데이터 F5: 주소가 바이트 단위로 지정되므로 다음 주소 201에는 하위에서 두 번째 바이트 F5가 저장된다.

주소 202 → 데이터 2A: 그다음 주소 202에는 하위에서 세 번째 바이트 2A가 저장된다.

주소 203 → 데이터 30: 마지막 주소 203에는 가장 상위인 네 번째 바이트 30이 저장된다.

주소 200, 201, 202, 203에 각각 67, F5, 2A, 30이 대응하므로 ①의 짝과 일치한다.

NoteBase

문 8. 데이터 전송 기법인 DMA(Direct Memory Access)에 대한 설명으로 옳지 않은 것은?

① DMA는 프로세서의 개입을 최소화하면서 주기억장치와 입출력장치 사이에 데이터를 전송하는 기술이다.

② 주기억장치와 입출력장치 사이에 대량의 데이터를 고속으로 전송 시, 인터럽트 방식이 DMA 방식보다 효율적이다.

③ 주기억장치와 입출력장치 사이에 DMA에 의한 데이터 전송 시, DMA 제어기는 버스 마스터(master)로 동작한다.

④ 단일 컴퓨터 시스템에 여러 개의 DMA 제어기가 존재할 수 있다.

정답 2번

간단한 해설

대량의 데이터를 전송할 때는 프로세서가 전송을 직접 처리하는 인터럽트 방식보다 **DMA 방식이 효율적**이다. DMA 제어기가 주기억장치와 입출력장치 사이의 전송을 맡아 프로세서의 처리 부담을 줄인다.

선지별 해설

① O: DMA는 프로세서가 데이터 하나하나를 옮기는 대신 **DMA 제어기가 직접 데이터를 전송**하는 방식이다. 프로세서는 전송 시작 전에 메모리 주소, 전송량 등을 설정하고, 전송이 끝나면 완료 통지를 받으므로 개입이 최소화된다.

② X: 인터럽트 방식의 입출력에서는 프로세서가 인터럽트 처리 루틴을 실행하여 데이터를 옮긴다. 대량 전송에서는 이러한 처리 부담이 커지므로 **인터럽트 방식보다 DMA 방식이 효율적**이다. DMA는 설정된 전송량만큼 제어기가 직접 데이터를 옮기며, 전송 완료를 알리는 데 인터럽트를 사용할 수 있다. 완료 통지에 인터럽트를 사용하더라도 데이터 전송 자체를 프로세서가 수행하는 것은 아니다.

③ O: 버스 마스터는 버스를 제어하여 주소와 읽기·쓰기 제어 신호를 내보내고 전송을 시작하는 주체이다. DMA 제어기는 버스 사용권을 얻은 뒤 **버스 마스터로 동작**하여 주기억장치와 입출력장치 사이의 데이터 전송을 수행한다.

④ O: 한 컴퓨터 시스템에 **여러 DMA 제어기를 둘 수 있다**. 각 제어기가 서로 다른 장치의 전송을 맡을 수 있으며, 같은 버스를 공유하는 제어기들이 동시에 사용을 요청하면 버스 중재를 통해 사용권을 정한다. 따라서 DMA 제어기가 반드시 하나여야 하는 것은 아니다.

NoteBase

문 9. PMBOK(Project Management Body of Knowledge)에서 제시하는 소프트웨어 프로젝트 관리 영역에 대한 설명으로 옳지 않은 것은?

① 프로젝트 일정 관리(time management)는 주어진 기간 내에 프로젝트를 완료하기 위한 활동에 대해 다룬다.

② 프로젝트 비용 관리(cost management)는 승인된 예산 내에서 프로젝트를 완료하기 위한 활동에 대해 다룬다.

③ 프로젝트 품질 관리(quality management)는 품질 요구를 만족하여 수행 목표를 달성하기 위한 활동에 대해 다룬다.

④ 프로젝트 조달 관리(procurement management)는 완성된 소프트웨어를 고객에게 전달하기 위한 활동에 대해 다룬다.

정답 4번

간단한 해설

④의 '완성된 소프트웨어를 고객에게 전달'은 조달 관리의 의미를 잘못 설명한 것이다. 조달 관리는 프로젝트 수행에 필요한 **제품·서비스 등을 외부에서 구매하거나 확보**하는 활동을 다룬다.

선지별 해설

① O: 프로젝트 일정 관리는 **정해진 기간 안에 프로젝트를 완료**하도록 작업의 순서와 소요 기간을 정하고 일정을 수립·통제하는 활동이다. 실제 진행 상황을 계획과 비교하고 지연이 생기면 일정을 조정한다.

② O: 프로젝트 비용 관리는 비용을 추정하고 예산을 수립하며 지출을 통제하여 **승인된 예산 안에서 프로젝트를 완료**하도록 하는 활동이다. 비용 초과 여부를 파악하고 필요한 조치를 취하는 것도 포함한다.

③ O: 프로젝트 품질 관리는 프로젝트와 산출물이 **정해진 품질 요구사항을 충족**하도록 하는 활동이다. 품질 기준을 정하고 작업 과정과 산출물을 점검하여 요구된 수행 목표를 달성하도록 한다.

④ X: 프로젝트 조달 관리는 프로젝트 수행에 필요한 제품·서비스·결과물을 **프로젝트 팀 외부에서 구매하거나 확보**하고 관련 계약을 관리하는 활동이다. 예를 들어 개발에 필요한 상용 소프트웨어 라이선스를 구매하거나 외부 업체에 시험 업무를 맡기는 것이 이에 해당한다. 따라서 '완성된 소프트웨어를 고객에게 전달하기 위한 활동'은 '프로젝트에 필요한 제품·서비스 등을 외부에서 조달하기 위한 활동'으로 고쳐야 한다.

NoteBase

문 10. 프로그램 구현 기법은 컴파일러를 이용한 기법, 인터프리터를 이용한 기법, 하이브리드(hybrid) 기법으로 구분된다. 이에 대한 설명으로 옳지 않은 것은?

① 하이브리드 기법에서는 인터프리터가 중간 언어로 번역된 프로그램을 해석하고 실행한다.

② 인터프리터를 이용한 기법에서는 고급 언어 프로그램을 명령문 단위로 하나씩 해석하여 바로 실행한다.

③ 반복문이 많은 프로그램의 실행에서 컴파일러를 이용한 기법이 인터프리터를 이용한 기법보다 효율적이다.

④ 인터프리터를 이용한 기법은 번역된 프로그램을 저장하기 위한 큰 기억 장소를 요구하는 단점이 있다.

정답 4번

간단한 해설

④의 '번역된 프로그램을 저장하기 위한 큰 기억 장소'는 **컴파일러 방식**에서 고려하는 단점이다. 순수 인터프리터 방식은 원시 프로그램을 해석하며 실행하므로, 전체를 번역한 목적 프로그램을 별도로 저장할 필요가 없다.

선지별 해설

① O: 하이브리드 기법은 원시 프로그램을 먼저 **중간 언어로 번역**한 뒤, 그 결과를 인터프리터가 해석하여 실행하는 방식이다. 컴파일 단계와 해석·실행 단계를 결합한다.

② O: 순수 인터프리터 방식은 고급 언어의 원시 프로그램을 **명령문 단위로 해석하면서 바로 실행**한다. 전체 프로그램을 기계어 목적 프로그램으로 미리 번역한 다음 실행하는 컴파일러 방식과 구별된다.

③ O: 컴파일러 방식은 실행 전에 프로그램을 번역해 두므로, 반복문을 실행할 때 이미 번역된 코드를 되풀이하여 실행한다. 순수 인터프리터 방식은 반복되는 명령문도 실행할 때마다 해석하는 부담이 있어, 일반적으로 반복문이 많은 프로그램에서는 **컴파일러 방식의 실행 효율이 더 높다**.

④ X: 순수 인터프리터 방식은 원시 프로그램 전체를 기계어 목적 프로그램으로 만들어 별도로 저장하지 않는다. 따라서 '번역된 프로그램을 저장하기 위한 큰 기억 장소를 요구한다'는 설명은 **컴파일러 방식의 단점**으로 고쳐야 한다. 이는 인터프리터 자체나 실행 중인 데이터에 필요한 기억 공간까지 없다는 뜻은 아니다.

NoteBase

문 11. 객체지향 프로그래밍에 대한 설명으로 옳지 않은 것은?

① 다형성(polymorphism)을 이용할 수 있다.

② 추상 클래스(abstract class)로부터 객체를 직접 생성할 수 없다.

③ 객체 간에는 메시지(message)를 통해 명령을 전달한다.

④ 상속(inheritance)이란 기존의 여러 클래스들을 조합하여 새로운 클래스를 만드는 기법이다.

정답 4번

간단한 해설

상속은 기존 클래스의 **속성과 메서드를 물려받아** 새로운 하위 클래스를 정의하는 기법이다. ④의 '여러 클래스들을 조합'한다는 표현은 상속의 핵심인 상위 클래스와 하위 클래스의 관계를 설명하지 못한다.

선지별 해설

① O: 다형성은 **같은 메시지나 메서드 호출에 객체의 종류에 따라 다르게 반응**하는 성질이다. 예를 들어 도형 객체에 그리기를 요청하면 원 객체는 원을, 사각형 객체는 사각형을 그리도록 구현할 수 있다.

② O: 추상 클래스는 하위 클래스들이 공유할 구조와 기능을 정의하는 데 쓰이며, **그 자체를 직접 인스턴스화할 수 없다**. 구체적인 하위 클래스가 필요한 기능을 구현하면 그 하위 클래스의 객체를 생성하여 사용한다.

③ O: 메시지는 한 객체가 다른 객체에 특정 동작을 요청하는 수단이다. 일반적으로 **상대 객체의 메서드를 호출**하는 방식으로 전달되며, 호출된 객체는 해당 메서드를 실행해 요청을 처리한다.

④ X: 상속은 기존 상위 클래스의 **속성과 메서드를 물려받아 하위 클래스를 정의**하는 기법이다. 하위 클래스는 물려받은 기능을 재사용하면서 기능을 추가하거나 메서드를 재정의할 수 있다. 다른 클래스의 객체를 구성 요소로 포함하여 기능을 조합하는 합성과 구별된다.

NoteBase

문 12. 시간 순서대로 제시된 다음의 시스템 운영 기록만을 이용하여 시스템의 가용성(availability)을 계산한 결과는?

(단위 : 시간)

가동시간	고장시간	가동시간	고장시간	가동시간	고장시간
8	1	7	2	9	3

① 80%

② 400%

③ 25%

④ 75%

정답 1번

간단한 해설

가용성은 **고장시간을 포함한 전체 시간 중 정상 가동시간의 비율**이다. 가동시간은 24시간, 고장시간은 6시간이므로 전체 30시간에 대한 가용성은 80%이다.

계산 과정

가용성은 관측한 전체 시간 중 시스템을 정상적으로 사용할 수 있었던 시간의 비율이다. 가동시간을 U, 고장시간을 D라 하면 가용성은 $A = \frac{U}{U+D} \times 100\%$이다. 분모에는 **가동시간과 고장시간을 모두 포함**한다.

표의 단위는 시간이다. 가동시간은 8 + 7 + 9 = 24시간이고, 고장시간은 1 + 2 + 3 = 6시간이다. 따라서 기록된 전체 시간은 24 + 6 = 30시간이다.

가용성은 $A = \frac{24}{30} \times 100\% = 80\%$이다. 이는 기록된 전체 시간의 80% 동안 시스템이 정상 가동되었다는 뜻이며, ①의 80%에 해당한다.

NoteBase

문 13. 다음에서 설명하는 기술은?

◯ 자동차를 기반으로 각종 정보를 주고받을 수 있는 자동차용 원격정보 서비스 기술
◯ 교통정보, 차량안전 및 보안, 차량진단, 생활정보 등의 서비스를 제공

① 텔레매틱스(Telematics)
② USN(Ubiquitous Sensor Network)
③ 증강현실(Augmented Reality)
④ 와이브로(WiBro)

정답 1번

간단한 해설

자동차를 통신망과 연결하여 교통정보·차량진단·안전 등의 서비스를 제공하는 기술은 **텔레매틱스**이다. 단서는 단순한 무선 접속이 아니라 **자동차용 원격정보 서비스**라는 점이다.

공통 해설

텔레매틱스(Telematics)는 **자동차에 통신 기술과 정보처리 기술을 결합**하여 차량과 외부 사이에 정보를 주고받고, 운전자와 차량에 필요한 서비스를 제공하는 기술이다.

지문의 교통정보와 생활정보는 운전자에게 필요한 정보를 제공하는 서비스이고, 차량안전·보안과 차량진단은 차량의 상태를 파악하고 관리하는 서비스이다. 이처럼 제공하는 정보의 종류가 달라도 **자동차를 기반으로 원격에서 정보를 주고받는 서비스**라는 공통점이 있으므로 텔레매틱스에 해당한다. 특정 통신망을 이용한다는 사실만으로 기술을 구분하는 것이 아니라, 자동차를 대상으로 어떤 서비스를 제공하는지를 기준으로 이해해야 한다.

NoteBase

문 14. 다음과 같은 가용 공간을 갖는 주기억장치에 크기가 각각 25 KB, 30 KB, 15 KB, 10 KB인 프로세스가 순차적으로 적재 요청된다. 최악적합(worst-fit) 배치전략을 사용할 경우 할당되는 가용 공간 시작주소를 순서대로 나열한 것은?

가용 공간 리스트	
시작주소	크기
w	30 KB
x	20 KB
y	15 KB
z	35 KB

① w → x → y → z

② x → y → z → w

③ y → z → w → x

④ z → w → x → y

정답 4번

간단한 해설

최악적합은 매번 **현재 가장 큰 가용 공간**에 프로세스를 배치한다. 처음에는 35 KB인 z에 25 KB를 할당하고, 이후에는 앞선 할당으로 줄어든 가용 공간의 크기를 기준으로 다음 위치를 고른다.

NoteBase

올바른 순서

z → w → x → y

z — 25 KB 요청: 최악적합은 요청을 수용할 수 있는 공간 중 **크기가 가장 큰 공간**을 선택한다. 표의 초기 가용 공간은 w가 30 KB, x가 20 KB, y가 15 KB, z가 35 KB이므로 z에 할당한다. z에는 35 − 25 = 10 KB가 남는다.

w — 30 KB 요청: 현재 가용 공간의 크기는 w가 30 KB, x가 20 KB, y가 15 KB, z의 남은 부분이 10 KB이다. 가장 큰 w에 할당하면 30 − 30 = 0 KB이므로 **w의 가용 공간은 모두 소진된다**.

x — 15 KB 요청: 남아 있는 가용 공간은 x의 20 KB, y의 15 KB, z의 남은 10 KB이다. 요청 크기와 정확히 같은 y가 있어도 최악적합은 **더 큰 x의 20 KB**를 선택한다. 할당 후 x에는 20 − 15 = 5 KB가 남는다.

y — 10 KB 요청: 현재 남은 공간은 x의 남은 5 KB, y의 15 KB, z의 남은 10 KB이다. 이 중 가장 큰 y에 할당하며, y에는 15 − 10 = 5 KB가 남는다.

NoteBase

문 15. 32K × 8비트 ROM 칩에 대한 설명으로 옳지 않은 것은?

① 이 ROM 칩 4개와 디코더(decoder)를 이용하여 128K × 8비트 ROM 모듈을 구현할 수 있다.

② 데이터 핀은 8개이다.

③ 워드 크기가 8비트인 컴퓨터 시스템에서만 사용된다.

④ 32,768개의 주소로 이루어진 주소 공간(address space)을 갖게 된다.

정답 3번

간단한 해설

③의 '8비트인 컴퓨터 시스템에서만 사용된다'는 제한이 잘못되었다. ROM 칩의 8비트는 **한 주소에서 읽어 내는 데이터의 폭**이며, 칩을 병렬로 연결하면 16비트 등 더 넓은 데이터를 읽는 시스템에도 사용할 수 있다.

선지별 해설

① O: 32K × 8비트 칩 4개로 주소 수를 늘리면 32K × 4 = 128K가 되고, 각 주소의 데이터 폭은 8비트로 유지된다. 하위 주소 15비트는 모든 칩에 공통으로 연결하고, 상위 주소 2비트를 2 대 4 디코더에 입력하여 **네 칩 중 하나만 선택**하도록 구성한다.

② O: '× 8비트'는 한 주소에 저장된 데이터를 읽을 때 8비트를 함께 출력한다는 뜻이다. 데이터 핀 하나가 1비트를 전달하므로 **데이터 출력 핀은 8개**이다. 주소 핀이나 칩 선택 핀은 데이터 핀에 포함하지 않는다.

③ X: 칩의 데이터 폭이 8비트라고 해서 컴퓨터의 워드 크기도 반드시 8비트여야 하는 것은 아니다. 예를 들어 칩 2개에 같은 주소를 입력하고, 각 칩의 출력을 16비트 데이터 버스의 하위 8비트와 상위 8비트에 각각 연결하면 32K × 16비트 ROM을 구성할 수 있다. 따라서 **워드 크기가 8비트보다 큰 시스템에서도 사용**할 수 있다.

④ O: 메모리 용량 표기에서 K는 1,024를 뜻하므로 32K는 32 × 1,024 = 32,768이다. 따라서 이 칩에는 **서로 다른 주소 32,768개**가 있으며, 각 주소에 8비트가 저장된다. 주소 수가 2^{15}개이므로 이를 구별하는 주소 비트는 15개이다.

NoteBase

문 16. 빅데이터에 대한 설명으로 옳지 않은 것은?

① 빅데이터의 특성을 나타내는 3V는 규모(Volume), 속도(Velocity), 가상화(Virtualization)를 의미한다.

② 빅데이터는 그림, 영상 등의 비정형 데이터를 포함한다.

③ 자연어 처리는 빅데이터 분석기술 중의 하나이다.

④ 시각화(visualization)는 데이터 분석 결과를 쉽게 이해할 수 있도록 표현하는 기술이다.

정답 1번

간단한 해설

빅데이터의 3V에서 데이터 형태의 폭넓음을 뜻하는 특성은 **다양성(Variety)**이다. ①의 '가상화(Virtualization)'를 '다양성(Variety)'으로 고쳐야 한다.

선지별 해설

① X: 빅데이터의 3V는 규모(Volume), 속도(Velocity), **다양성(Variety)**이다. 규모는 데이터의 양이 방대함을, 속도는 데이터가 빠르게 생성·수집·처리됨을, 다양성은 데이터의 형태와 출처가 다양함을 뜻한다. 가상화(Virtualization)는 3V에 포함되지 않는다.

② O: 빅데이터는 표처럼 일정한 구조를 갖춘 정형 데이터뿐 아니라 그림·영상·음성 등의 **비정형 데이터도 포함**한다. 비정형 데이터는 미리 정해진 행과 열의 구조로 표현하기 어려운 데이터이다.

③ O: 자연어 처리는 사람이 사용하는 언어를 컴퓨터가 분석하고 처리하는 기술이다. 대량의 상품 후기에서 의견이나 감정을 파악하는 것처럼 **텍스트 데이터의 의미를 분석**하는 데 활용되므로 빅데이터 분석기술에 해당한다.

④ O: 시각화는 데이터를 그래프·도표·지도 등의 시각적 형태로 나타내는 기술이다. 분석 결과의 **패턴·추세·관계를 쉽게 파악**하도록 돕는다.

NoteBase

문 17. 다음 Java 프로그램의 출력 결과는?

```
class Foo {
    public int a = 3;
    public void addValue(int i) {
        a = a + i;
        System.out.println("Foo : "+ a + " " );
    }
    public void addFive() {
        a += 5;
        System.out.println("Foo : "+ a + " " );
    }
}

class Bar extends Foo {
    public int a = 8;
    public void addValue(double i) {
        a = a + (int)i;
        System.out.println("Bar : "+ a + " " );
    }
    public void addFive() {
        a += 5;
        System.out.println("Bar : "+ a + " " );
    }
}

public class Test {
    public static void main(String [] args) {
        Foo f = new Bar();
        f.addValue(1);
        f.addFive();
    }
}
```

① Foo : 4
Foo : 9

② Bar : 9
Foo : 8

③ Foo : 4
Bar : 13

④ Bar : 9
Bar : 14

정답 3번

간단한 해설

f.addValue(1)은 Foo의 int 매개변수 메서드를 실행하여 **Foo의 a를 3에서 4로** 바꾼다. f.addFive()는 **Bar에서 재정의한 메서드**를 실행하므로, 별도로 유지된 Bar의 a에 5를 더해 13을 출력한다.

NoteBase

계산 과정

초기 상태: Foo f = new Bar()에서 f의 선언 타입은 Foo이고, 실제 생성된 객체의 타입은 Bar이다. 이 객체에는 Foo에 선언된 a와 Bar에 선언된 a가 **서로 다른 필드로 존재**하며, 초기값은 각각 3과 8이다. Bar에 같은 이름의 필드가 있어도 Foo의 필드가 없어지거나 두 값이 합쳐지지 않는다.

f.addValue(1): 메서드의 오버로딩은 참조 변수의 선언 타입과 인수 타입을 기준으로 컴파일 시 결정된다. f는 Foo 타입이고 1은 int이므로 Foo의 addValue(int)가 선택된다. Bar의 addValue(double)은 매개변수 타입이 달라 **오버라이딩이 아니라 오버로딩**이며, Foo의 addValue(int)를 대체하지 않는다. 따라서 Foo의 메서드에서 Foo에 선언된 a를 사용하여 3 + 1 = 4로 바꾸고, 첫 줄에 'Foo : 4'를 출력한다. 이때 Bar의 a는 여전히 8이다.

f.addFive(): Bar의 addFive()는 Foo의 addFive()와 이름 및 매개변수 목록이 같아 이를 오버라이딩한다. 이처럼 재정의된 인스턴스 메서드는 **실제 객체 타입인 Bar**의 구현이 실행된다. Bar의 메서드 안에서 사용하는 a는 Bar에 선언된 필드이므로, 앞서 바뀐 Foo의 a 값 4가 아니라 Bar의 a 값 8에 5를 더한다. 따라서 8 + 5 = 13이 되어 둘째 줄에 'Bar : 13'을 출력한다.

출력 결과: 첫 줄은 'Foo : 4', 둘째 줄은 'Bar : 13'이며, ③의 출력과 일치한다.

NoteBase

문 18. 다음 부울식을 간략화한 것은?

$AB + A'C + ABD' + A'CD' + BCD'$

① $A'C + BC$
② $AB + BC$
③ $AB + A'C$
④ $A'CD' + BCD'$

정답 3번

간단한 해설

흡수법칙으로 ABD'와 $A'CD'$를 없앨 수 있다. 남은 BCD'도 **A가 1이면 AB에, 0이면 A'C에 포함**되므로 간략식은 $AB + A'C$이다.

계산 과정

부울식에서 +는 OR, 곱은 AND, 작은따옴표는 NOT을 뜻하며 각 변수는 0 또는 1의 값을 갖는다.

1. 흡수법칙 적용: **이미 포함된 조건에 다른 조건을 덧붙인 항은 제거**할 수 있다. 흡수법칙 $X + XY = X$에 따라 $AB + ABD' = AB$, $A'C + A'CD' = A'C$이다. 따라서 주어진 식은 $AB + A'C + BCD'$로 줄어든다.

2. 남은 항 제거: $BCD' = 1$이면 B와 C는 1이고 D는 0이다. 이때 A가 1이면 $AB = 1$이고, A가 0이면 $A'C = 1$이다. 즉, **어느 경우에도 앞의 두 항 중 하나가 이미 1**이므로 BCD'는 식의 출력에 영향을 주지 않는다. 따라서 $AB + A'C + BCD' = AB + A'C$이다.

3. 간략화 결과: 주어진 부울식은 $AB + A'C$로 간략화되며, 이는 ③의 식과 일치한다.

NoteBase

문 19. 다음은 속성(attribute) A, B, C, D와 4개의 투플(tuple)로 구성되고 두 개의 함수 종속 AB → C, A → D를 만족하는 릴레이션을 나타낸다. ㉠과 ㉡에 들어갈 수 있는 속성 값이 옳게 짝지어진 것은? (단, A 속성의 도메인은 {a1, a2, a3, a4}이고, D 속성의 도메인은 {d1, d2, d3, d4, d5}이다)

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	㉠
㉡	b1	c1	d3
a4	b1	c4	d4

① ㉠: d1, ㉡: a1

② ㉠: d1, ㉡: a2 또는 a3

③ ㉠: d5, ㉡: a2 또는 a4

④ ㉠: d4, ㉡: a4

정답 2번

간단한 해설

A → D는 **A 값이 같으면 D 값도 같아야 한다**는 뜻이므로, A가 a1인 두 행의 D 값은 d1로 일치해야 한다. ㉡에 a1이나 a4를 넣으면 기존 행과 함수 종속이 충돌하므로, 가능한 값은 **a2 또는 a3**이다.

NoteBase

빈칸별 해설

함수 종속은 결정자 값이 같은 두 투플에서 종속되는 속성 값도 같아야 한다는 제약이다. AB → C에서는 A와 B가 모두 같을 때 C도 같아야 하고, A → D에서는 A가 같을 때 D도 같아야 한다. 결정자 값이 서로 다르면 종속되는 값이 같아도 되고 달라도 된다.

㉠: d1. 첫째 행과 둘째 행은 A가 모두 a1이다. A → D에 따라 **B 값이 달라도 D 값은 같아야 하므로**, 둘째 행의 ㉠은 첫째 행의 D 값인 d1이다. 두 행은 B가 b1과 b2로 다르므로 AB → C는 이 두 행의 C 값이 같을 것을 요구하지 않는다.

㉡: a2 또는 a3. 셋째 행의 D는 d3이다. ㉡이 a1이면 첫째 행과 A는 같은데 D가 d1과 d3으로 달라 A → D를 위반한다. ㉡이 a4이면 넷째 행과 A는 같은데 D가 d3과 d4로 달라 A → D를 위반한다. 이때는 B도 b1로 같지만 C가 c1과 c4로 달라 AB → C도 위반한다. 도메인에 남은 **a2와 a3은 어느 쪽을 넣어도 다른 행과 A가 겹치지 않으므로** 두 함수 종속을 모두 만족한다.

㉠이 d1이고 ㉡이 a2 또는 a3인 조합은 ②에 해당한다.

NoteBase

문 20. 서브넷 마스크(subnet mask)가 255.255.255.192인 서브넷의 IP 주소에서 호스트 식별자(hostid)의 비트 수는?

① 5

② 6

③ 7

④ 8

정답 2번

간단한 해설

서브넷 마스크에서 **0인 비트가 호스트 식별자**에 해당한다. 255.255.255.192의 마지막 부분은 이진수로 11000000이므로 호스트 식별자는 6비트이다.

계산 과정

IPv4 주소는 32비트이며, 서브넷 마스크의 각 비트는 IP 주소의 같은 위치에 대응한다. 마스크가 1인 위치는 네트워크 식별자에, **0인 위치는 호스트 식별자**에 해당한다.

마스크의 각 부분은 8비트이다. 255는 이진수로 11111111이고, 192는 128 + 64이므로 이진수로 11000000이다. 따라서 255.255.255.192를 이진수로 나타내면 11111111.11111111.11111111.11000000이다.

앞의 세 부분에는 1이 24개, 마지막 부분에는 1이 2개 있으므로 네트워크 식별자는 24 + 2 = 26비트이다. 호스트 식별자는 나머지 32 − 26 = 6비트이며, 이는 마스크 끝의 0 여섯 개에 대응한다. 따라서 호스트 식별자의 비트 수는 ②의 6이다.

NoteBase