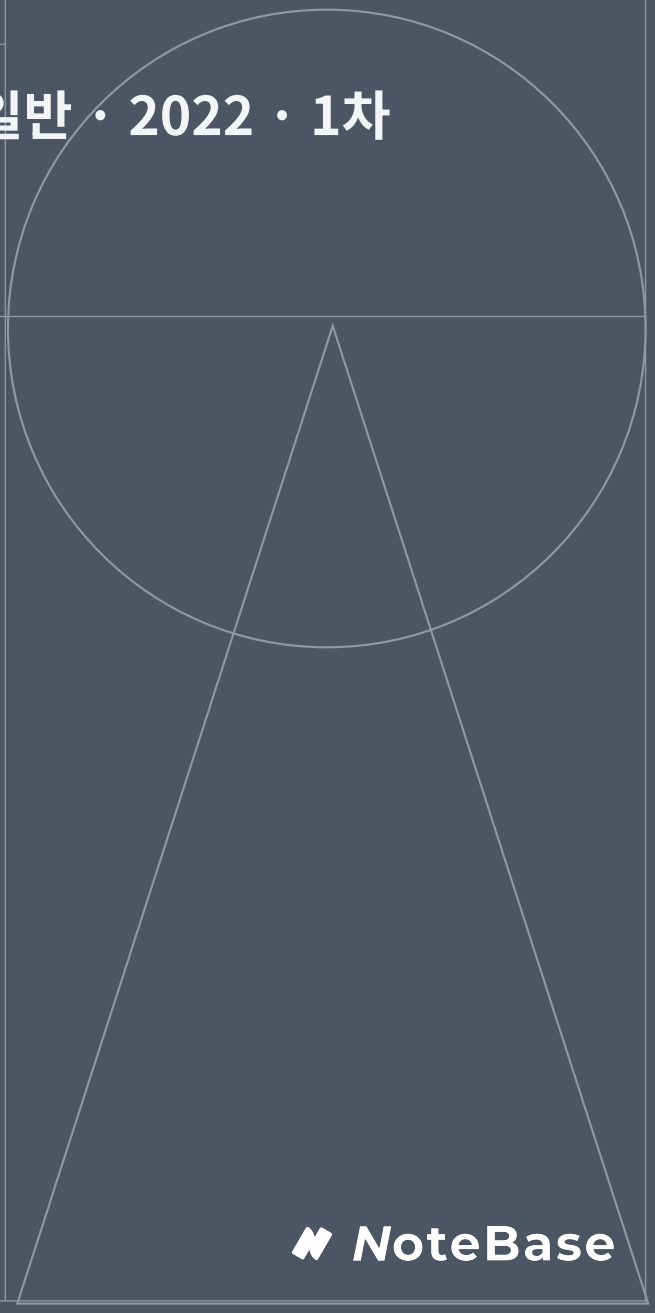
기출OX 해설집

공무원/해경/순경 · 컴퓨터일반 · 2022 · 1차



NoteBase

기출OX 해설집

공무원/해경/순경 · 컴퓨터일반 · 2022 · 1차

공무원/해경/순경 · 컴퓨터일반 · 2022 · 1차

문 1. 클라우드 컴퓨팅 서비스 모델과 이에 대한 설명이 가장 옳게 짝지어진 것은?

<보기>
㉠ 응용소프트웨어 개발에 필요한 개발 요소들과 실행 환경을 제공하는 서비스 모델로, 사용자는 원하는 응용소프트웨어를 개발할 수 있으나 운영체제나 하드웨어에 대한 제어는 서비스 제공자에 의해 제한된다.
㉡ 응용소프트웨어 및 관련 데이터는 클라우드에 호스팅 되고 사용자는 웹 브라우저 등의 클라이언트를 통해 접속하여 응용소프트웨어를 사용할 수 있다.
㉢ 사용자 필요에 따라 가상화된 서버, 스토리지, 네트워크 등의 인프라 자원을 제공한다.

① ㉠ - SaaS, ㉡ - PaaS, ㉢ - IaaS
② ㉠ - PaaS, ㉡ - IaaS, ㉢ - SaaS
③ ㉠ - PaaS, ㉡ - SaaS, ㉢ - IaaS
④ ㉠ - IaaS, ㉡ - SaaS, ㉢ - PaaS

정답 3번

출제 포인트

클라우드 컴퓨팅 서비스 모델인 PaaS, SaaS, IaaS를 사용자가 제공받는 자원의 범위와 직접 관리하는 계층에 따라 구별하는 문제이다.

핵심 개념

PaaS(Platform as a Service): 응용소프트웨어를 개발하고 실행하는 데 필요한 플랫폼과 실행 환경을 제공하는 서비스이다. 사용자는 응용소프트웨어 개발에 집중하고 운영체제와 하드웨어는 주로 제공자가 관리한다.
SaaS(Software as a Service): 제공자가 운영하는 완성된 응용소프트웨어를 인터넷을 통해 이용하는 서비스이다. 사용자는 보통 웹 브라우저 등의 클라이언트로 접속한다.
IaaS(Infrastructure as a Service): 가상화된 서버, 스토리지, 네트워크 등 컴퓨팅 인프라를 제공하는 서비스이다.
핵심 구별은 개발·실행 플랫폼이면 PaaS, 완성된 소프트웨어를 사용하면 SaaS, 인프라 자원을 제공받으면 IaaS라는 점이다.

NoteBase

선지별 해설

① X
㉠의 'SaaS'와 ㉡의 'PaaS'가 틀렸다. ㉠은 개발 요소와 실행 환경을 제공하므로 PaaS이고, ㉡은 클라우드에 호스팅된 응용소프트웨어를 브라우저 등으로 사용하므로 SaaS이다. ㉢의 IaaS는 옳다. 올바른 연결은 '㉠-PaaS, ㉡-SaaS, ㉢-IaaS'이다.

② X
㉡의 'IaaS'와 ㉢의 'SaaS'가 틀렸다. ㉡은 완성된 응용소프트웨어를 이용하는 SaaS이고, ㉢은 서버·스토리지·네트워크 같은 인프라 자원을 제공하는 IaaS이다. ㉠의 PaaS는 옳다. 올바른 연결은 '㉠-PaaS, ㉡-SaaS, ㉢-IaaS'이다.

③ O
㉠은 응용소프트웨어 개발 플랫폼을 제공하므로 PaaS, ㉡은 호스팅된 응용소프트웨어를 사용하게 하므로 SaaS, ㉢은 가상화된 인프라 자원을 제공하므로 IaaS이다. 세 연결이 모두 옳다.

④ X
㉠의 'IaaS'와 ㉢의 'PaaS'가 틀렸다. ㉠은 개발·실행 환경을 제공하는 PaaS이고, ㉢은 서버·스토리지·네트워크를 제공하는 IaaS이다. ㉡의 SaaS는 옳다. 올바른 연결은 '㉠-PaaS, ㉡-SaaS, ㉢-IaaS'이다.

오답 포인트

서비스 이름의 약자만 외우면 ㉠의 '개발 요소와 실행 환경'을 소프트웨어 제공으로 오인하거나, ㉢의 가상화 자원을 플랫폼으로 오인하기 쉽다. 이 문항의 함정은 제공 대상이 플랫폼인지, 완성된 소프트웨어인지, 기반 인프라인지를 서로 바꾸어 놓은 데 있다.

다시 나오면 이렇게 판단

문장에서 제공 대상을 먼저 찾는다. 개발·실행 환경은 PaaS, 브라우저로 이용하는 완성된 응용소프트웨어는 SaaS, 서버·스토리지·네트워크는 IaaS로 판단한다.

암기 포인트

PaaS는 개발 플랫폼, SaaS는 완성 소프트웨어, IaaS는 서버·스토리지·네트워크 인프라이다.

NoteBase

문 2. 전자계산기의 중앙처리장치(CPU)는 4가지 단계를 반복적으로 거치면서 동작을 행한다. 4가지 단계에 가장 속하지 않는 것은?

① Fetch Cycle
② Interrupt Cycle
③ Branch Cycle
④ Execute Cycle

정답 3번

출제 포인트

이 문제는 CPU의 명령 사이클을 구성하는 인출, 간접, 실행, 인터럽트 사이클을 구별하는 문제이다.

핵심 개념

명령 사이클은 CPU가 한 명령어를 처리하기 위해 반복하는 일련의 과정이다. 인출 사이클에서는 기억장치에서 명령어를 가져오고, 간접 사이클에서는 간접 주소 지정에 필요한 유효주소를 구하며, 실행 사이클에서는 명령어가 지시한 연산을 수행한다. 인터럽트 요청이 있으면 인터럽트 사이클에서 현재 상태를 보존하고 인터럽트 처리로 넘어간다. Branch는 분기 명령의 실행 결과로 프로그램의 흐름을 바꾸는 동작이지, 명령 사이클의 독립된 단계가 아니다.

선지별 해설

① O — Fetch Cycle은 기억장치에서 다음에 실행할 명령어를 가져와 명령어 레지스터에 저장하는 인출 사이클이므로 명령 사이클에 속한다.
② O — Interrupt Cycle은 인터럽트 요청을 검사하고, 요청이 있으면 복귀에 필요한 현재 상태를 저장한 뒤 인터럽트 서비스 루틴으로 제어를 넘기는 사이클이므로 명령 사이클에 속한다.
③ X — Branch Cycle이라는 독립된 명령 사이클 단계는 없다. 틀린 표현은 ‘Branch Cycle’이며, 올바르게는 ‘Indirect Cycle’이다. Branch는 분기 명령을 실행할 때 프로그램 카운터를 분기 주소로 변경하는 실행 동작이다.
④ O — Execute Cycle은 해독된 명령어가 요구하는 산술·논리 연산, 자료 이동, 분기 등의 실제 동작을 수행하는 실행 사이클이므로 명령 사이클에 속한다.

오답 포인트

③의 Branch는 CPU가 실제로 수행하는 중요한 동작이어서 하나의 사이클처럼 보이지만, 분기는 Execute Cycle에서 수행되는 명령 동작이다. 명령 사이클의 네 종류를 묻는 경우 Branch가 아니라 Indirect가 포함된다.

NoteBase

다시 나오면 이렇게 판단

CPU 명령 사이클의 단계인지 판단할 때는 '명령어 인출 → 필요하면 유효주소 계산 → 명령 실행 → 필요하면 인터럽트 처리'의 흐름에 대응하는지 확인한다. 이에 대응하는 명칭은 Fetch, Indirect, Execute, Interrupt이다.

암기 포인트

명령 사이클 4종은 인출(Fetch)·간접(Indirect)·실행(Execute)·인터럽트(Interrupt)이며, Branch는 실행 사이클 안의 분기 동작이다.

NoteBase

문 3. ARP(Address Resolution Protocol)에 대한 설명으로 가장 옳은 것은?

① 호스트의 MAC주소에 해당하는 IP주소를 구할 수 있다.

② ARP 요청은 해당 네트워크상의 모든 호스트와 라우터에게 유니캐스트(unicast)된다.

③ ARP 응답은 ARP 요청을 전송한 요청자와 모든 호스트에게 브로드캐스트(broadcast)된다.

④ ARP는 인증 기능이 없으므로 위조된 정보를 사용할 수 있다.

정답 4번

출제 포인트

ARP의 주소 변환 방향, 요청·응답의 전송 방식, 인증 기능의 유무를 구별하는 문제이다.

핵심 개념

ARP는 같은 네트워크에서 목적지의 IP 주소에 대응하는 MAC 주소를 알아내는 프로토콜이다. 요청자는 대상 MAC 주소를 모르므로 ARP 요청을 브로드캐스트하고, 해당 IP 주소를 가진 호스트는 보통 요청자에게 ARP 응답을 유니캐스트한다. ARP 자체에는 응답 정보의 진위를 확인하는 인증 기능이 없다.

선지별 해설

① X

틀린 표현은 'MAC주소에 해당하는 IP주소'이다. ARP는 반대로 IP 주소에 해당하는 MAC 주소를 구한다. MAC 주소로 IP 주소를 구하는 기능은 전통적으로 RARP에 해당한다.

② X

틀린 표현은 '유니캐스트된다'이다. 대상의 MAC 주소를 아직 모르므로 ARP 요청은 같은 브로드캐스트 도메인의 호스트들이 받을 수 있도록 브로드캐스트된다.

③ X

틀린 표현은 '요청자와 모든 호스트에게 브로드캐스트된다'이다. 일반적인 ARP 응답은 요청을 보낸 호스트의 MAC 주소를 알고 있으므로 그 요청자에게 유니캐스트된다.

④ O

ARP에는 응답의 진위를 검증하는 인증 기능이 없으므로 공격자가 위조된 IP-MAC 대응 정보를 보내 ARP 캐시를 오염시킬 수 있다.

NoteBase

오답 포인트

①은 주소 변환 방향을 뒤집었고, ②와 ③은 ARP 요청과 응답의 전송 방식을 서로 바꾼 함정이다. 요청은 목적지 MAC 주소를 모르므로 브로드캐스트하고, 응답은 요청자의 MAC 주소를 알기 때문에 일반적으로 유니캐스트한다.

다시 나오면 이렇게 판단

ARP는 'IP 주소로 MAC 주소 찾기'라는 방향을 먼저 확정한다. 이어서 처음에는 대상 MAC 주소를 모르므로 요청은 브로드캐스트, 요청 패킷에서 요청자의 MAC 주소를 알게 된 뒤 보내는 응답은 일반적으로 유니캐스트라고 판단한다.

암기 포인트

ARP는 IP→MAC, 요청은 브로드캐스트, 응답은 유니캐스트, 자체 인증은 없다.

NoteBase

문 4. Access Time이 빠른 순서로 나열할 때 가장 옳은 것은?

<보기>
㉠ Cache Memory
㉡ Register
㉢ Main Memory
㉣ Magnetic Disk

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

정답 2번

출제 포인트

이 문제는 기억장치 계층에서 접근 시간이 짧아 데이터 접근이 빠른 순서를 묻는 문제이다.

올바른 순서

㉡ Register → ㉠ Cache Memory → ㉢ Main Memory → ㉣ Magnetic Disk
레지스터는 CPU 내부에서 즉시 사용하는 값을 보관하고, 캐시는 주기억장치 접근을 줄이기 위해 자주 쓰는 데이터를 임시 보관한다. 그다음이 주기억장치이며, 기계적 동작을 수반하는 자기 디스크가 가장 느리다.

선지별 해설

① X — '㉠ Cache Memory → ㉡ Register'의 선후가 틀렸다. 레지스터가 캐시 메모리보다 접근 속도가 빠르므로 올바르게는 ㉡ → ㉠ → ㉢ → ㉣이다.
② O — Register → Cache Memory → Main Memory → Magnetic Disk 순으로, 접근 속도가 빠른 장치부터 정확히 나열했다.
③ X — '㉢ Main Memory'가 가장 앞에 온 부분이 틀렸다. 주기억장치는 레지스터와 캐시 메모리보다 느리므로 올바르게는 ㉡ → ㉠ → ㉢ → ㉣이다.
④ X — '㉢ Main Memory → ㉡ Register → ㉠ Cache Memory'의 순서가 틀렸다. 레지스터와 캐시 메모리가 주기억장치보다 빠르므로 올바르게는 ㉡ → ㉠ → ㉢ → ㉣이다.

오답 포인트

레지스터와 캐시 메모리를 바꾸어 생각하거나, 용량이 큰 장치를 빠른 장치로 오해하기 쉽다. 이 문항의 기준은 용량이 아니라 접근 시간이며, 접근 시간이 짧을수록 빠르다.

NoteBase

다시 나오면 이렇게 판단

CPU에 가까운 저장 계층부터 확인한다. CPU 내부의 레지스터, CPU와 주기억장치 사이의 캐시, 주기억장치, 보조기억장치인 자기 디스크 순으로 판단한다.

암기 포인트

빠른 순서는 레지스터 → 캐시 → 주기억장치 → 자기 디스크이다.

NoteBase

문 5. 비선점(Non-Preemptive) 스케줄링에 가장 해당하지 않는 것은?

① SRT(Shortest Remaining Time)

② FCFS(First Come First Served)

③ SJF(Shortest Job First)

④ HRN(Highest Response ratio Next)

정답 1번

출제 포인트

이 문제는 CPU 스케줄링 알고리즘을 선점형과 비선점형으로 구분할 수 있는지를 묻는다.

핵심 개념

선점형 스케줄링은 실행 중인 프로세스의 CPU를 운영체제가 회수하여 다른 프로세스에 배정할 수 있는 방식이다. 비선점형 스케줄링은 한 프로세스가 CPU를 배정받으면 종료되거나 대기 상태로 전환될 때까지 다른 프로세스가 CPU를 빼앗을 수 없는 방식이다. SRT는 남은 실행 시간이 더 짧은 프로세스가 도착하면 현재 프로세스를 중단시키는 선점형이고, FCFS·SJF·HRN은 기본적으로 비선점형이다.

선지별 해설

① O — SRT는 남은 실행 시간이 가장 짧은 프로세스를 우선하며, 더 짧은 잔여 시간을 가진 프로세스가 도착하면 실행 중인 프로세스를 선점한다. 따라서 비선점 스케줄링에 해당하지 않는다.
② X — ‘FCFS가 비선점 스케줄링에 해당하지 않는다’는 판단이 틀리다. 올바르게는 도착한 순서대로 CPU를 배정하고 실행 중인 프로세스를 강제로 중단하지 않는 비선점형이다.
③ X — ‘SJF가 비선점 스케줄링에 해당하지 않는다’는 판단이 틀리다. 올바르게는 준비 큐에서 실행 시간이 가장 짧은 작업을 선택한 뒤 그 작업의 실행을 보장하는 비선점형이며, 이를 선점형으로 만든 방식이 SRT이다.
④ X — ‘HRN이 비선점 스케줄링에 해당하지 않는다’는 판단이 틀리다. 올바르게는 응답률이 가장 높은 작업을 선택하여 실행시키는 비선점형이다.

오답 포인트

SJF와 SRT가 모두 짧은 작업을 우선한다는 점 때문에 혼동하기 쉽다. 핵심 차이는 새 프로세스가 도착했을 때 현재 실행을 중단할 수 있는지이며, SRT만 잔여 시간을 다시 비교하여 선점한다.

다시 나오면 이렇게 판단

알고리즘 이름보다 실행 중인 CPU를 회수할 수 있는지를 먼저 확인한다. 남은 시간이 더 짧은 프로세스의 도착으로 현재 프로세스가 중단되면 SRT이므로 선점형이고, FCFS·SJF·HRN은 비선점형으로 판단한다.

NoteBase

암기 포인트

SJF는 비선점, SRT는 SJF의 선점형 버전이다. FCFS·SJF·HRN은 비선점형이다.

NoteBase

문 6. 다음에서 운영체제에 대한 설명으로 옳은 것만을 고른 것은?

<보기>
㉠ 운영체제는 중앙처리장치, 주기억장치, 보조기억장치, 주변장치 등의 컴퓨터 자원을 할당 및 관리하는 시스템 소프트웨어이다.
㉡ 스풀링(spooling)은 CPU와 입출력 장치의 속도 차이를 줄이기 위해 주기억장치의 일부분을 버퍼처럼 사용하는 것이다.
㉢ 비선점(non-preemptive) 방식의 CPU 스케줄링 기법은 CPU를 사용하고 있는 현재의 프로세스가 종료된 후 다른 프로세스에 CPU를 할당하는데 대표적으로 RR(Round Robin) 스케줄링 기법이 있다.
㉣ 가상메모리(virtual memory)는 디스크와 같은 보조기억장치에 가상의 공간을 만들어 주기억장치처럼 활용하도록 하여 실제 주기억장치의 물리적 공간보다 큰 주소 공간을 제공한다.

① ㉠, ㉡
② ㉠, ㉢
③ ㉠, ㉣
④ ㉢, ㉣

정답 3번

출제 포인트

운영체제의 자원 관리 기능과 스풀링, CPU 스케줄링, 가상메모리의 핵심 개념을 구별하는 문제이다.

핵심 개념

운영체제는 CPU, 기억장치, 입출력장치 등의 자원을 관리하는 시스템 소프트웨어이다.
스풀링은 입출력할 데이터를 디스크와 같은 보조기억장치에 임시로 저장하여 CPU와 입출력장치가 독립적으로 작업하게 하는 방식이다. 주기억장치의 일부를 임시 저장 공간으로 쓰는 것은 버퍼링에 해당한다.
선점형 스케줄링은 실행 중인 프로세스에서 CPU를 회수할 수 있으며, RR은 시간 할당량이 끝나면 CPU를 회수하는 대표적인 선점형 방식이다.
가상메모리는 보조기억장치를 활용하여 물리적 주기억장치보다 큰 논리적 주소 공간을 제공한다.

NoteBase

선지별 해설

① X — ㉠은 옳지만 ㉡은 틀리므로 옳은 설명만 고른 조합이 아니다. ㉡의 '주기억장치의 일부분을 버퍼처럼 사용'이 틀린 표현이며, 스풀링은 디스크와 같은 보조기억장치에 입출력 데이터를 임시 저장하는 방식이다. 따라서 ㉠, ㉡이 아니라 ㉠, ㉣이 옳다.
② X — ㉠은 옳지만 ㉢은 틀리므로 옳은 설명만 고른 조합이 아니다. ㉢의 '비선점 방식의 대표적인 기법이 RR'이라는 부분이 틀렸으며, RR은 정해진 시간 할당량이 끝나면 CPU를 회수하는 선점형 스케줄링 기법이다. 따라서 ㉠, ㉢이 아니라 ㉠, ㉣이 옳다.
③ O — ㉠은 운영체제의 자원 관리 기능을 정확히 설명하고, ㉣은 보조기억장치를 활용하여 물리적 주기억장치보다 큰 주소 공간을 제공하는 가상메모리를 정확히 설명한다.
④ X — ㉣은 옳지만 ㉢은 틀리므로 옳은 설명만 고른 조합이 아니다. RR은 비선점형이 아니라 선점형 방식이므로 ㉢, ㉣이 아니라 ㉠, ㉣이 옳다.

오답 포인트

㉡은 스풀링과 버퍼링의 저장 위치를 바꾸어 놓았고, ㉢은 RR을 비선점형으로 바꾸어 놓았다. '보조기억장치에 대기 작업 저장'은 스풀링, '시간 할당량이 끝나면 CPU 회수'는 선점형 RR이라는 점이 핵심 함정이다.

다시 나오면 이렇게 판단

스풀링은 보조기억장치, 버퍼링은 주기억장치의 임시 저장 영역으로 구분한다. 스케줄링은 실행 중 CPU를 강제로 회수할 수 있는지를 확인하며, 시간 할당량을 사용하는 RR은 선점형으로 판단한다.

암기 포인트

스풀링은 디스크, RR은 선점, 가상메모리는 보조기억장치로 주소 공간 확장이다.

NoteBase

문 7. UNIX 시스템에서 커널의 수행 기능에 해당하는 것으로 가장 옳지 않은 것은?

① 프로세스 관리

② 기억장치 관리

③ 입출력 관리

④ 명령어 해독

정답 4번

출제 포인트

UNIX 커널이 담당하는 운영체제의 자원 관리 기능과 CPU 제어장치의 명령어 처리 기능을 구별하는 문제이다.

핵심 개념

커널은 운영체제의 핵심 부분으로, 프로세스·기억장치·입출력 장치 등 시스템 자원을 관리한다. 반면 명령어 해독은 CPU의 제어장치가 명령어 레지스터의 명령을 분석하여 수행할 동작을 결정하는 과정이다.

선지별 해설

① O — 프로세스 생성·종료, 스케줄링, 프로세스 간 통신 등 프로세스 관리는 커널의 기능이다.
② O — 주기억장치의 할당과 회수, 가상 메모리 운영 등 기억장치 관리는 커널의 기능이다.
③ O — 장치 드라이버와 인터럽트 등을 통해 입출력 장치를 제어하고 관리하는 것은 커널의 기능이다.
④ X — '명령어 해독'은 커널의 자원 관리 기능이 아니다. 올바르게는 CPU의 제어장치가 인출된 명령어의 연산 코드 등을 해독한다.

오답 포인트

④의 '명령어'라는 말 때문에 운영체제가 사용자 명령을 처리하는 기능과 혼동하기 쉽다. 이 선지의 명령어 해독은 프로그램 명령어를 CPU 내부에서 해석하는 단계이므로 커널 기능이 아니라 제어장치의 기능이다.

다시 나오면 이렇게 판단

프로세스·메모리·입출력처럼 시스템 자원을 배분하고 통제하는 기능은 커널로 판단한다. 인출·해독·실행처럼 기계 명령어의 처리 주기에 속하는 기능은 CPU의 제어장치와 연산장치 영역으로 판단한다.

암기 포인트

커널은 자원 관리, 명령어 해독은 CPU 제어장치이다.

NoteBase

문 8. 파이프라이닝(pipelining) 기법이 적용된 중앙처리 장치(CPU)에서의 파이프라인 해저드(pipeline hazard) 종류와 대응 방법을 바르게 짝지은 것만을 모두 고른 것은?

<보기>
㉠ 데이터 해저드(data hazard)
→ 데이터 전방 전달(data forwarding)
㉡ 구조적 해저드(structural hazard)
→ 부족한 자원의 추가
㉢ 제어 해저드(control hazard)
→ 분기 예측(branch prediction)

① ㉠, ㉡
② ㉠, ㉢
③ ㉡, ㉢
④ ㉠, ㉡, ㉢

정답 4번

출제 포인트

파이프라인 해저드의 세 종류인 데이터 해저드, 구조적 해저드, 제어 해저드와 각각의 대표적인 대응 방법을 구별하는 문제이다.

핵심 개념

데이터 해저드: 앞 명령어의 연산 결과가 아직 준비되지 않았는데 뒤 명령어가 그 값을 필요로 하는 충돌이다. 데이터 전방 전달로 결과를 레지스터에 기록할 때까지 기다리지 않고 필요한 단계로 직접 보낼 수 있다.
구조적 해저드: 여러 명령어가 같은 시점에 동일한 하드웨어 자원을 요구하여 생기는 충돌이다. 부족한 자원을 추가하거나 자원을 분리하여 완화할 수 있다.
제어 해저드: 분기 명령의 결과가 확정되기 전에 다음에 실행할 명령어를 결정하기 어려워 생기는 충돌이다. 분기 예측으로 다음 실행 경로를 미리 추정하여 완화할 수 있다.
따라서 ㉠, ㉡, ㉢은 모두 바르게 짝지어져 있다.

NoteBase

선지별 해설

① X — ㉠과 ㉡의 대응은 맞지만 ㉢도 올바른 짝이므로, '㉠, ㉡'만 고른 범위가 틀렸다. 올바르게는 ㉠, ㉡, ㉢을 모두 골라야 한다.
② X — ㉠과 ㉢의 대응은 맞지만 ㉡도 올바른 짝이므로, '㉠, ㉢'만 고른 범위가 틀렸다. 올바르게는 ㉠, ㉡, ㉢을 모두 골라야 한다.
③ X — ㉡과 ㉢의 대응은 맞지만 ㉠도 올바른 짝이므로, '㉡, ㉢'만 고른 범위가 틀렸다. 올바르게는 ㉠, ㉡, ㉢을 모두 골라야 한다.
④ O — 데이터 해저드에는 데이터 전방 전달, 구조적 해저드에는 부족한 자원의 추가, 제어 해저드에는 분기 예측이 각각 대응하므로 세 항목을 모두 고른 ④가 옳다.

오답 포인트

각 대응 방법이 해저드를 완전히 제거하는지가 아니라 해당 해저드를 줄이는 대표 방법인지를 판단해야 한다. 특히 분기 예측은 예측 실패 가능성이 있어도 제어 해저드의 대표적인 대응 방법이므로 ㉢을 제외하면 안 된다.

다시 나오면 이렇게 판단

값의 의존 관계는 전방 전달, 하드웨어 자원의 충돌은 자원 추가·분리, 분기 이후 경로의 불확실성은 분기 예측으로 연결한다. 그런 다음 각 항목을 개별 판정하고 옳은 항목을 빠짐없이 포함한 선지를 고른다.

암기 포인트

데이터-전방 전달, 구조-자원 추가, 제어-분기 예측으로 연결한다.

NoteBase

문 9. 유비쿼터스 컴퓨팅에 대한 설명으로 가장 옳지 않은 것은?

① 노매딕(nomadic) 컴퓨팅은 현실 세계와 가상 화면을 결합하여 보여주는 기술이다.

② 감지 컴퓨팅은 컴퓨터가 센서 등을 이용하여 사용자의 행위 또는 주변 환경을 인식하여 필요 정보를 제공하는 기술이다.

③ 퍼베이시브(pervasive) 컴퓨팅은 컴퓨터가 도처에 편재되도록 하는 기술이다.

④ 웨어러블(wearable) 컴퓨팅은 컴퓨터 착용을 통해 컴퓨터를 인간 몸의 일부로 여길 수 있도록 하는 기술이다.

정답 1번

출제 포인트

유비쿼터스 컴퓨팅의 구현 형태인 노매딕·감지·퍼베이시브·웨어러블 컴퓨팅의 개념을 구별하는 문제이다.

핵심 개념

노매딕 컴퓨팅: 사용자가 이동하면서도 장소에 구애받지 않고 정보 시스템과 네트워크를 이용하는 컴퓨팅이다.
감지 컴퓨팅: 센서로 사용자의 행위나 주변 환경을 인식하고 상황에 맞는 정보를 제공하는 컴퓨팅이다.
퍼베이시브 컴퓨팅: 컴퓨팅 장치와 기능이 생활 공간 곳곳에 스며들어 편재하는 컴퓨팅이다.
웨어러블 컴퓨팅: 컴퓨터를 신체나 의복에 착용하여 일상적으로 이용하는 컴퓨팅이다.
현실 세계와 가상 정보를 결합해 보여 주는 기술은 증강현실이다.

선지별 해설

① X — '현실 세계와 가상 화면을 결합하여 보여준다'는 부분이 틀렸다. 이는 증강현실에 대한 설명이다. 노매딕 컴퓨팅은 이동 중에도 장소의 제약 없이 컴퓨팅 자원과 네트워크를 이용하는 기술이다.
② O — 감지 컴퓨팅은 센서 등을 이용해 사용자의 행위와 주변 환경을 인식하고, 그 상황에 필요한 정보를 제공한다.
③ O — 퍼베이시브 컴퓨팅은 컴퓨터와 컴퓨팅 기능이 일상 공간 도처에 편재하도록 하는 기술이다.
④ O — 웨어러블 컴퓨팅은 컴퓨터를 신체나 의복에 착용하여 인간의 몸과 밀접하게 결합해 사용하는 기술이다.

오답 포인트

①은 '현실과 가상의 결합'이라는 익숙한 표현을 노매딕 컴퓨팅에 붙인 함정이다. 현실 화면 위에 가상 정보를 겹쳐 보여 주는 것은 증강현실이고, 노매딕 컴퓨팅의 핵심은 사용자의 이동성과 장소 독립성이다.

다시 나오면 이렇게 판단

용어를 핵심 키워드와 연결한다. 이동하며 접속하면 노매딕, 센서로 상황을 인식하면 감지, 도처에 컴퓨팅 기능이 존재하면 퍼베이시브, 몸에 착용하면 웨어러블, 현실과 가상 정보를 결합하면 증강현실이다.

NoteBase

암기 포인트

노매딕은 이동, 감지는 센서, 퍼베이시브는 편재, 웨어러블은 착용, 증강현실은 현실과 가상의 결합이다.

NoteBase

문 10. 캐시메모리의 매핑(mapping) 방법이 아닌 것은?

① direct mapping
② indirect mapping
③ associative mapping
④ set-associative mapping

정답 2번

출제 포인트

캐시메모리에 주기억장치 블록을 배치하는 표준 매핑 방식 세 가지를 구별하는 문제이다.

핵심 개념

직접 매핑(direct mapping)은 각 주기억장치 블록이 들어갈 캐시 라인이 하나로 정해지는 방식이다.
연관 매핑(associative mapping)은 주기억장치의 블록을 캐시의 어느 라인에나 배치할 수 있는 방식이다. 완전 연관 매핑이라고도 한다.
집합 연관 매핑(set-associative mapping)은 캐시를 여러 집합으로 나누고, 블록은 정해진 집합 안의 어느 라인에나 배치할 수 있는 방식이다.

선지별 해설

① O — direct mapping은 표준 캐시 매핑 방식이다. 각 블록이 배치될 캐시 라인이 하나로 고정된다.
② X — indirect mapping은 캐시메모리의 표준 매핑 방식이 아니다. 올바른 표준 분류는 direct mapping, associative mapping, set-associative mapping이다.
③ O — associative mapping은 표준 캐시 매핑 방식이다. 블록을 캐시의 어느 라인에나 배치할 수 있다.
④ O — set-associative mapping은 표준 캐시 매핑 방식이다. 블록은 하나의 집합에 대응하고 그 집합 안의 어느 라인에나 배치될 수 있다.

오답 포인트

②의 indirect mapping을 direct mapping과 짝을 이루는 정식 용어로 착각하기 쉽다. 캐시 매핑의 표준 세 분류에는 indirect mapping이 포함되지 않는다.

다시 나오면 이렇게 판단

배치 가능 범위를 기준으로 구별한다. 한 라인만 가능하면 직접 매핑, 모든 라인이 가능하면 연관 매핑, 정해진 집합 안의 여러 라인이 가능하면 집합 연관 매핑이다.

NoteBase

암기 포인트

캐시 매핑의 세 방식은 직접·연관·집합 연관이며, 간접 매핑은 해당하지 않는다.

NoteBase

문 11. Unix에서 <보기>의 작업을 수행하는 명령어들을 나열한 것 중 가장 옳은 것은?

<보기>
㉠ 디렉터리 내의 파일 목록 확인
㉡ 현재 실행 중인 프로세스들의 정보를 확인
㉢ 현재 실행 중인 특정 프로세스 종료
㉣ 파일 또는 디렉터리의 디스크 사용량 확인

① ㉠ ls ㉡ netstat ㉢ bg ㉣ wc
② ㉠ dir ㉡ ps ㉢ bg ㉣ wc
③ ㉠ ls ㉡ netstat ㉢ kill ㉣ du
④ ㉠ ls ㉡ ps ㉢ kill ㉣ du

정답 4번

출제 포인트

Unix의 파일 목록, 프로세스 조회·종료, 디스크 사용량 확인 명령어를 작업별로 정확히 연결하는 문제이다.

핵심 개념

ls: 디렉터리 안의 파일과 하위 디렉터리 목록을 확인한다.
ps: 현재 실행 중인 프로세스의 정보를 확인한다.
kill: 프로세스 식별자(PID)를 지정해 해당 프로세스에 종료 등의 신호를 보낸다.
du: 파일이나 디렉터리가 차지하는 디스크 사용량을 확인한다.
netstat은 네트워크 연결·통계 확인, bg는 중지된 작업의 백그라운드 실행 재개, wc는 행·단어·바이트 수 계산에 쓰인다.

NoteBase

선지별 해설

① X
㉠ ls는 맞지만, ㉡ netstat은 프로세스 목록이 아니라 네트워크 상태를 확인하고, ㉢ bg는 특정 프로세스를 종료하지 않으며, ㉣ wc는 디스크 사용량을 확인하지 않는다. 올바르게는 ㉡ ps, ㉢ kill, ㉣ du이다.

② X
㉡ ps는 맞지만, 이 문항에서 ㉠ 파일 목록 확인의 표준적인 대응은 ls이고, ㉢ bg는 종료가 아니라 백그라운드 실행 재개, ㉣ wc는 디스크 사용량이 아니라 행·단어·바이트 수 계산 명령이다. 올바르게는 ㉠ ls, ㉢ kill, ㉣ du이다.

③ X
㉠ ls, ㉢ kill, ㉣ du는 맞지만, ㉡ netstat은 네트워크 상태 확인 명령이다. 현재 실행 중인 프로세스 정보 확인에는 ps를 사용한다.

④ O
㉠ ls는 파일 목록, ㉡ ps는 프로세스 정보, ㉢ kill은 특정 프로세스 종료, ㉣ du는 디스크 사용량 확인에 각각 대응한다.

오답 포인트

명령어가 모두 Unix에서 볼 수 있는 형태라는 점만 보고 고르면 틀리기 쉽다. 특히 netstat과 ps는 각각 네트워크 상태와 프로세스 상태, bg와 kill은 각각 작업 재개와 프로세스 종료, wc와 du는 각각 내용의 개수와 디스크 사용량을 다룬다는 차이가 핵심이다.

다시 나오면 이렇게 판단

대상을 먼저 구분한다. 파일 목록은 ls, 프로세스 목록은 ps, 프로세스 종료는 kill, 저장 공간 사용량은 du로 연결한다.

암기 포인트

목록 ls, 프로세스 ps, 종료 kill, 용량 du이다.

NoteBase

문 12. 다음 중 <보기>의 Java 프로그램 실행 결과로 가장 옳은 것은?

<보기>

```
class CodeRunner {
public static void main(String[] args) {
A obj = new C();
obj.f();
obj.g();
}
}
class A {
public void f() {System.out.print("1 ");}
public static void g() {System.out.print("2 ");}
}
class B extends A {
public void f() {System.out.print("3 ");}
}
class C extends B {
public static void g() {System.out.print("4 ");}
}
```

① 3 2

② 3 1

③ 1 4

④ 1 2

정답 1번

출제 포인트

Java에서 인스턴스 메서드는 실행 시 실제 객체의 타입에 따라 동적 바인딩되지만, 정적 메서드는 컴파일 시 참조 변수의 선언 타입에 따라 결정된다는 차이를 묻는 문제이다.

NoteBase

핵심 개념

인스턴스 메서드: 오버라이딩될 수 있으며, 호출할 메서드는 실제 객체의 타입을 기준으로 결정된다. 이 코드에서 실제 객체는 C이고, C가 f()를 재정의하지 않았으므로 가장 가까운 상위 클래스 B의 f()가 실행되어 3을 출력한다.
정적 메서드: 클래스에 속하며 오버라이딩이 아니라 숨김이 적용된다. 호출할 메서드는 참조 변수의 선언 타입을 기준으로 결정되므로, A형 변수 obj를 통한 obj.g()는 A.g()를 호출하여 2를 출력한다. C.g()의 4는 출력되지 않는다.

선지별 해설

① O — obj.f()는 B의 f()를 호출하여 3을, obj.g()는 A의 정적 메서드 g()를 호출하여 2를 출력하므로 실행 결과는 3 2이다.
② X — 틀린 부분은 두 번째 출력값 1이다. g()는 A.g()를 호출하므로 올바른 실행 결과는 3 2이다.
③ X — 첫 번째 출력값 1과 두 번째 출력값 4가 모두 틀리다. f()는 실제 객체의 상속 관계에서 B.f()가 선택되고, g()는 참조 변수의 선언 타입 A에 따라 A.g()가 선택되므로 올바른 실행 결과는 3 2이다.
④ X — 틀린 부분은 첫 번째 출력값 1이다. f()는 A.f()가 아니라 오버라이딩된 B.f()를 호출하므로 올바른 실행 결과는 3 2이다.

오답 포인트

obj의 선언 타입이 A라는 사실을 모든 메서드 호출에 똑같이 적용하거나, 실제 객체가 C라는 이유로 모든 메서드를 C에서 찾는 것이 함정이다. f()는 동적 바인딩으로 B.f()가 선택되지만, static인 g()는 A형 참조를 기준으로 A.g()가 선택된다.

다시 나오면 이렇게 판단

먼저 호출 대상 메서드에 static이 있는지 확인한다. 인스턴스 메서드는 실제 객체 타입에서 위로 올라가며 가장 가까운 재정의 메서드를 찾고, 정적 메서드는 참조 변수의 선언 타입에 속한 메서드를 찾는다.

암기 포인트

인스턴스 메서드는 실제 객체, 정적 메서드는 선언 타입을 기준으로 결정된다.

NoteBase

문 13. 다음 중 <보기1>의 테이블 R에 대해 <보기2>의 SQL을 수행한 결과로 가장 옳은 것은?

<보기1>

A	B
3	1
2	4
3	2
2	5
3	3
1	5

<보기2>

```
SELECT SUM(B) FROM R
GROUP BY A
HAVING COUNT(B) = 2;
```

① 9
② 5
③ 6
④ 2

정답 1번

출제 포인트

GROUP BY로 만든 그룹에 HAVING 조건을 적용한 뒤, 남은 그룹의 집계값을 구할 수 있는지를 묻는 문제이다.

조건 정리

A=3인 그룹의 B 값은 1, 2, 3으로 3개이고, A=2인 그룹의 B 값은 4, 5로 2개이며, A=1인 그룹의 B 값은 5로 1개이다. HAVING COUNT(B)=2를 만족하는 그룹만 남겨야 한다.

사용할 원리

GROUP BY A는 A 값이 같은 행들을 하나의 그룹으로 묶는다. HAVING은 이렇게 만들어진 그룹을 집계 조건으로 필터링하므로, B 값의 개수가 2개인 A=2 그룹만 남는다. SELECT SUM(B)는 남은 그룹에서 B의 합계를 반환한다.

NoteBase

계산 과정

1. A=3 그룹은 COUNT(B)=3이므로 제외된다.
2. A=2 그룹은 COUNT(B)=2이므로 남는다.
3. A=1 그룹은 COUNT(B)=1이므로 제외된다.
4. 남은 A=2 그룹의 B 값 4와 5를 합하면 4+5=9이다.

선지 판단

① O — HAVING 조건을 만족하는 A=2 그룹의 B 합계가 4+5=9이므로 옳다.
② X — 5는 A=1 그룹의 B 값이거나 A=2 그룹의 개별 B 값일 뿐, 조건을 만족한 그룹의 합계가 아니다. 올바른 결과는 9이다.
③ X — 6은 A=3 그룹의 B 합계 1+2+3이지만, 이 그룹은 COUNT(B)=3이므로 HAVING COUNT(B)=2를 만족하지 않는다. 올바른 결과는 9이다.
④ X — 2는 조건을 만족하는 그룹의 행 개수 COUNT(B)이지 SELECT 절이 요구한 SUM(B)가 아니다. 올바른 결과는 9이다.

오답 포인트

HAVING의 비교값 2를 그대로 결과로 고르거나, HAVING으로 제외된 A=3 그룹의 합계 6을 고르는 것이 함정이다. HAVING은 출력값을 정하는 절이 아니라 집계가 끝난 그룹을 거르는 절이다.

다시 나오면 이렇게 판단

GROUP BY로 그룹을 나누고, 각 그룹의 집계값으로 HAVING 조건을 판정한 다음, 살아남은 그룹에 SELECT의 집계함수를 적용하는 순서로 판단한다.

암기 포인트

GROUP BY는 그룹 생성, HAVING은 그룹 필터링, SELECT의 집계함수는 최종 결과 계산이다.

NoteBase

문 14. 다음 중 <보기>에서 설명하는 기술로 가장 옳은 것은?

<보기>
- 클라이언트의 요구에 대한 응답 시간을 줄일 수 있다.
- 외부 인터넷과 연결된 트래픽을 줄일 수 있다.
- 최근 호출된 객체의 사본을 저장한다.

① DNS server
② NAT server
③ Router
④ Proxy server

정답 4번

출제 포인트

클라이언트의 요청을 대신 처리하고 최근 요청된 객체의 사본을 저장하여 응답 시간과 외부 트래픽을 줄이는 프록시 서버의 캐시 기능을 묻는 문제이다.

핵심 개념

프록시 서버는 클라이언트와 외부 서버 사이에서 요청과 응답을 중계하는 서버이다. 자주 또는 최근 요청된 웹 객체를 캐시에 저장해 두었다가 같은 요청에 재사용하면 응답 시간이 짧아지고 외부 인터넷으로 나가는 트래픽도 감소한다.
DNS 서버는 도메인 이름을 IP 주소로 변환하고, NAT 서버는 사설 주소와 공인 주소를 변환하며, 라우터는 서로 다른 네트워크 사이에서 패킷의 전달 경로를 선택한다.

선지별 해설

① X — DNS server는 도메인 이름을 IP 주소로 변환하는 이름 해석을 담당한다. 틀린 표현은 최근 호출된 객체의 사본을 저장해 응답 시간과 외부 트래픽을 줄인다는 기능을 DNS의 기능으로 본 점이며, 올바르게는 캐시 기능을 사용하는 Proxy server의 설명이다.
② X — NAT server는 사설 IP 주소와 공인 IP 주소를 변환하여 내부망과 외부망의 통신을 가능하게 한다. 틀린 표현은 객체 사본의 저장과 재사용을 NAT의 기능으로 본 점이며, 올바르게는 Proxy server의 캐시 기능이다.
③ X — Router는 목적지 IP 주소와 라우팅 정보를 바탕으로 네트워크 사이에서 패킷을 전달한다. 틀린 표현은 최근 호출된 객체를 저장하여 클라이언트에 재제공하는 기능을 Router의 기능으로 본 점이며, 올바르게는 Proxy server의 기능이다.
④ O — Proxy server는 클라이언트의 요청을 대신 중계하며, 최근 호출된 객체의 사본을 캐시에 저장해 재사용할 수 있다. 따라서 클라이언트의 응답 시간이 줄고 외부 인터넷과 연결된 트래픽도 감소한다.

NoteBase

오답 포인트

‘외부 인터넷 연결’이라는 표현만 보고 NAT나 Router를 고르기 쉽다. 그러나 ‘최근 호출된 객체의 사본을 저장한다’는 문구가 결정적이며, 이는 주소 변환이나 경로 선택이 아니라 프록시 캐시를 뜻한다.

다시 나오면 이렇게 판단

객체의 사본을 저장하고 같은 요청에 재사용하여 응답 시간과 외부 트래픽을 줄인다는 설명이면 프록시 서버의 캐시 기능으로 판단한다. 도메인 변환은 DNS, 주소 변환은 NAT, 패킷 경로 선택은 Router로 구분한다.

암기 포인트

DNS는 이름 변환, NAT는 주소 변환, Router는 경로 선택, Proxy는 요청 중계와 캐시이다.

NoteBase

문 15. 다음 중 네트워크 장비에 대한 설명으로 가장 옳지 않은 것은?

① 라우터(router)는 데이터 전송을 위한 최선의 경로를 설정한다.

② 브리지(bridge)는 한 네트워크 세그먼트에서 들어온 데이터를 그의 물리적 주소에 관계없이 무조건 다른 세그먼트로 전달한다.

③ 스위치(switch)는 보안(security) 및 트래픽(traffic) 관리 기능도 제공할 수 있다.

④ 허브(hub)는 전달받은 신호를 그와 연결된 모든 노드들에 전달한다.

정답 2번

출제 포인트

라우터, 브리지, 스위치, 허브가 데이터를 전달할 때 사용하는 기준과 전달 범위를 구별하는 문제이다.

핵심 개념

라우터는 IP 주소와 라우팅 정보를 바탕으로 서로 다른 네트워크 사이의 경로를 선택한다. 브리지는 MAC 주소를 학습하여 필요한 세그먼트로 프레임을 선별 전달한다. 스위치는 다수의 포트를 가진 브리지처럼 MAC 주소를 기준으로 프레임을 전달하며 보안 및 트래픽 관리 기능을 제공할 수 있다. 허브는 수신 신호를 목적지와 관계없이 다른 모든 포트로 내보낸다.

선지별 해설

① O — 라우터는 라우팅 테이블과 경로 선택 기준을 이용하여 목적지 네트워크까지의 적절한 경로를 정한다.
② X — 틀린 표현은 '물리적 주소에 관계없이 무조건 다른 세그먼트로 전달한다'이다. 브리지는 프레임의 MAC 주소를 학습하고 확인하여 필요한 세그먼트로 선별 전달한다. 목적지 MAC 주소를 아직 알지 못하거나 브로드캐스트 프레임인 경우 등에는 여러 포트로 전달할 수 있지만, 모든 데이터를 항상 무조건 전달하는 것은 아니다.
③ O — 관리형 스위치는 포트 보안, VLAN, 접근 제어, QoS 등의 기능으로 보안과 트래픽을 관리할 수 있다.
④ O — 허브는 물리 계층 장비로서 한 포트에서 받은 신호를 목적지 주소를 판별하지 않고 연결된 다른 모든 포트로 전달한다.

오답 포인트

②의 '무조건'이 함정이다. 브리지는 허브처럼 모든 프레임을 항상 전체 전송하는 장비가 아니라 MAC 주소를 기준으로 세그먼트 간 트래픽을 걸러 낸다.

다시 나오면 이렇게 판단

장비별 판단 기준을 확인한다. IP 주소와 경로 선택은 라우터, MAC 주소와 선별 전달은 브리지·스위치, 주소 판단 없이 전체 포트로 신호를 보내는 것은 허브이다.

NoteBase

암기 포인트

라우터는 IP·경로, 브리지와 스위치는 MAC·선별, 허브는 전체 전달이다.

NoteBase

문 16. 시큐어 코딩(Secure Coding) 기법으로 가장 옳지 않은 것은?

① 화이트리스트 방식으로 허용된 확장자만 업로드를 허용한다.

② 입력 화면 폼(Form) 작성시 POST 방식보다 GET 방식을 사용한다.

③ 사용자로부터 입력받은 스크립트 관련 문자열을 필터링 하여 변환한다.

④ 인자화된 질의문(parameterized query)을 사용한다.

정답 2번

출제 포인트

이 문제는 파일 업로드, 입력값 처리, 데이터베이스 질의에서 취약점을 줄이는 시큐어 코딩 기법을 구별하는 문제이다.

핵심 개념

시큐어 코딩은 소프트웨어를 설계·구현할 때 입력값 검증, 안전한 데이터 전송, 질의문 분리 등의 방법으로 보안 취약점이 생기지 않도록 하는 개발 기법이다.
파일 업로드는 허용 목록 방식으로 제한하고, 스크립트 입력은 필터링·인코딩하며, 데이터베이스 질의는 인자화하여 공격 문자열이 명령으로 해석되지 않게 한다. 폼 전송에서 GET은 입력값을 URL에 노출하므로 민감한 입력 처리에 적합하지 않으며, POST를 사용하더라도 전송 내용 보호에는 HTTPS가 별도로 필요하다.

선지별 해설

① O — 업로드 가능한 확장자를 허용 목록으로 정하고 그 밖의 확장자를 차단하면 실행 파일이나 스크립트 파일의 무분별한 업로드를 줄일 수 있다.
② X — 틀린 표현은 ‘POST 방식보다 GET 방식을 사용한다’이다. GET은 입력값이 URL의 질의 문자열에 포함되어 브라우저 기록, 로그 등에 남을 수 있으므로 민감한 입력 폼에는 POST 방식을 사용하는 것이 적절하다. 다만 POST 자체가 내용을 암호화하는 것은 아니므로 HTTPS를 함께 사용해야 한다.
③ O — 사용자가 입력한 스크립트 관련 문자열을 검증하고 필터링·인코딩하면 입력값이 실행 가능한 스크립트로 해석되는 것을 막아 크로스 사이트 스크립팅(XSS) 위험을 줄일 수 있다.
④ O — 인자화된 질의문은 SQL 문장 구조와 사용자 입력값을 분리하므로 입력값이 SQL 명령으로 실행되는 것을 막아 SQL 삽입 공격을 예방한다.

오답 포인트

②에서 GET과 POST를 단순한 전송 방식의 차이로만 보면 틀리기 쉽다. GET은 입력값이 URL에 드러날 수 있다는 점이 핵심이며, POST를 쓰면 자동으로 암호화된다고 이해해서도 안 된다.

NoteBase

다시 나오면 이렇게 판단

사용자 입력이 명령이나 코드로 해석되는지, 민감한 값이 URL에 노출되는지, 허용되지 않은 파일이 업로드될 수 있는지를 확인한다. 허용 목록, 출력 인코딩, 인자화된 질의문은 안전한 방향이고 민감한 폼 입력을 GET으로 보내는 것은 안전하지 않은 방향이다.

암기 포인트

업로드는 허용 목록, XSS는 필터링·인코딩, SQL 삽입은 인자화된 질의문, 민감한 폼 전송은 POST와 HTTPS이다.

NoteBase

문 17. 다음 중 RAID(Redundant Array of Inexpensive Disks)에 대한 설명으로 가장 옳지 않은 것은?

① RAID-0은 디스크 스트라이핑(Disk Striping) 방식으로 중복 저장과 오류 검출 및 교정이 없는 방식이다.

② RAID-1은 디스크 미러링(Disk Mirroring) 방식으로 높은 신뢰도를 갖는다.

③ RAID-4는 데이터를 비트(bit) 단위로 여러 디스크에 분할하여 저장하는 방식이며, 별도의 패리티(parity) 디스크를 사용한다.

④ RAID-5는 별도의 패리티(parity) 디스크 대신 모든 디스크에 패리티(parity) 정보를 나누어 기록하는 방식이다.

정답 3번

출제 포인트

RAID 0, 1, 4, 5를 데이터 분할 방식과 패리티 배치 방식으로 구별하는 문제이다.

핵심 개념

RAID 0: 데이터를 블록 단위로 여러 디스크에 분산하는 스트라이핑 방식이며 중복 정보가 없다.
RAID 1: 같은 데이터를 둘 이상의 디스크에 복제하는 미러링 방식이다.
RAID 4: 블록 단위 스트라이핑을 사용하고 패리티를 전용 디스크 한 곳에 저장한다.
RAID 5: 블록 단위 스트라이핑을 사용하고 패리티를 모든 디스크에 분산하여 저장한다.

선지별 해설

① O — RAID 0은 디스크 스트라이핑만 사용하므로 중복 저장 정보와 패리티에 의한 오류 검출·복구 기능이 없다.
② O — RAID 1은 동일한 데이터를 여러 디스크에 복제하는 미러링 방식이어서 한 디스크에 장애가 발생해도 복제본으로 데이터를 유지할 수 있으므로 신뢰도가 높다.
③ X — 틀린 표현은 '비트(bit) 단위'이다. RAID 4는 데이터를 블록 단위로 여러 디스크에 분할하고 별도의 전용 패리티 디스크를 사용하는 방식이다. 비트 단위 스트라이핑은 RAID 2, 바이트 단위 스트라이핑과 전용 패리티의 결합은 RAID 3의 특징이다.
④ O — RAID 5는 전용 패리티 디스크를 두지 않고 패리티 블록을 모든 디스크에 분산하여 기록하므로 특정 패리티 디스크에 작업이 집중되는 문제를 줄인다.

오답 포인트

③은 '별도의 패리티 디스크'라는 RAID 4의 특징과 '비트 단위 분할'이라는 RAID 2의 특징을 한 문장에 섞었다. RAID 4와 RAID 5는 모두 블록 단위로 데이터를 나누지만, 패리티가 전용 디스크에 모이면 RAID 4이고 여러 디스크에 분산되면 RAID 5이다.

NoteBase

다시 나오면 이렇게 판단

먼저 데이터 분할 단위가 블록인지 확인하고, 다음으로 패리티가 없는지, 전용 디스크에 있는지, 모든 디스크에 분산되는지를 확인한다. 패리티 없음은 RAID 0, 전용 패리티와 블록 스트라이핑은 RAID 4, 분산 패리티와 블록 스트라이핑은 RAID 5로 판단한다.

암기 포인트

0은 스트라이핑만, 1은 미러링, 4는 블록·전용 패리티, 5는 블록·분산 패리티이다.

NoteBase

문 18. 다음 중 소프트웨어 규모를 예측하기 위한 기능점수(function point)를 산정할 때 고려할 사항으로 가장 옳지 않은 것은?

① 원시 코드 라인 수(Line Of Code)

② 외부입력(External Input)

③ 외부조회(External inQuiry)

④ 내부논리파일(Internal Logical File)

정답 1번

출제 포인트

기능점수법이 소프트웨어의 구현 코드량이 아니라 사용자에게 제공되는 기능을 기준으로 규모를 산정한다는 점을 묻는 문제이다.

핵심 개념

기능점수(Function Point)는 사용자가 인식하는 기능을 입력·출력·조회·내부논리파일·외부연계파일의 다섯 유형으로 나누어 소프트웨어 규모를 측정하는 방식이다. 원시 코드 라인 수(LOC)는 구현된 소스 코드의 줄 수를 세는 별도의 규모 측정 지표이다.

선지별 해설

① X — 원시 코드 라인 수(Line Of Code)는 소스 코드의 물리적 길이를 측정하는 지표이므로 기능점수 산정의 기능 유형에 포함되지 않는다. 틀린 표현은 '원시 코드 라인 수'이며, 올바르게는 기능점수의 다섯 유형인 외부입력·외부출력·외부조회·내부논리파일·외부연계파일 가운데 하나를 고려한다.
② O — 외부입력(External Input)은 시스템 경계 밖에서 들어와 내부논리파일을 유지하거나 시스템 동작을 제어하는 데이터·제어 입력으로, 기능점수 산정 요소이다.
③ O — 외부조회(External Inquiry)는 입력에 대응하여 단순 조회 결과를 제공하되 내부논리파일을 변경하지 않는 기능으로, 기능점수 산정 요소이다.
④ O — 내부논리파일(Internal Logical File)은 사용자가 식별할 수 있고 해당 시스템 내부에서 유지·관리되는 논리적 데이터 집합으로, 기능점수 산정 요소이다.

오답 포인트

기능점수와 LOC가 모두 소프트웨어 규모를 나타내므로 혼동하기 쉽다. 그러나 ①의 '원시 코드 라인 수'는 구현 결과인 코드량을 세는 지표이고, 기능점수는 사용자 관점의 기능을 세는 지표라는 차이가 핵심이다.

NoteBase

다시 나오면 이렇게 판단

선지가 입력·출력·조회 또는 논리파일처럼 사용자가 인식하는 기능을 나타내면 기능점수 요소로 판단한다. 특정 프로그래밍 언어로 작성된 코드의 줄 수를 뜻하면 LOC이므로 기능점수 요소에서 제외한다.

암기 포인트

기능점수는 EI·EO·EQ·ILF·EIF의 다섯 기능, LOC는 코드 줄 수이다.

NoteBase

문 19. 다음 중 캐시기억장치 교체 알고리즘에 대한 설명으로 가장 옳지 않은 것은?

① LRU는 최근에 가장 오랫동안 사용되지 않았던 블록을 교체하는 방법이다.

② FIFO는 캐시에 적재된 후 가장 오래된 블록을 먼저 교체하는 방법이다.

③ LFU는 참조 횟수를 기록함으로써 가장 많이 참조된 블록을 교체하는 방법이다.

④ Random은 사용 횟수와 무관하게 임의로 블록을 교체하는 방법이다.

정답 3번

출제 포인트

캐시기억장치의 교체 알고리즘이 어떤 기준으로 교체할 블록을 선택하는지 구별하는 문제이다.

핵심 개념

LRU는 가장 오랫동안 참조되지 않은 블록을 교체한다. FIFO는 캐시에 가장 먼저 들어온 블록을 교체한다. LFU는 참조 횟수가 가장 적은 블록을 교체한다. Random은 정해진 사용 이력 기준 없이 임의의 블록을 교체한다.

선지별 해설

① O — LRU는 최근 사용 시점을 기준으로 가장 오랫동안 사용되지 않은 블록을 교체하므로 옳다.
② O — FIFO는 적재 순서를 기준으로 캐시에 가장 먼저 적재되어 가장 오래 머문 블록을 먼저 교체하므로 옳다.
③ X — 틀린 표현은 '가장 많이 참조된 블록'이다. LFU는 참조 횟수를 기록하여 가장 적게 참조된 블록을 교체한다.
④ O — Random은 참조 횟수나 최근 사용 시점 같은 사용 이력을 교체 기준으로 삼지 않고 블록을 임의로 선택하므로 옳다.

오답 포인트

③은 LFU가 참조 횟수를 기록한다는 앞부분은 맞지만, 교체 대상의 방향을 '가장 많이'로 뒤집었다. LFU의 L은 Least이므로 교체 대상은 참조 횟수가 가장 적은 블록이다.

다시 나오면 이렇게 판단

알고리즘마다 판단 기준과 교체 대상을 짝지어 확인한다. LRU는 최근성 중 가장 오래 미사용, FIFO는 적재 순서 중 가장 먼저 적재, LFU는 빈도 중 최소 참조, Random은 임의 선택이다.

암기 포인트

LRU는 오래 미사용, FIFO는 먼저 적재, LFU는 최소 빈도, Random은 임의 선택이다.

NoteBase

문 20. 다음 중 <보기>에서 설명하는 C프로그램의 실행 결과로 가장 옳은 것은?

< 보 기 >

```
#include <stdio.h>

int a = 10;
int b = 20;
int c = 30;

void func(void)
{
    static int a = 100;
    int b = 200;

    a++;
    b++;
    c = a;
}

int main(void)
{
    func( );
    func( );

    printf("a = %d, b = %d, c = %d \n", a, b, c);
}
```

① a = 10, b = 20, c = 30

② a = 10, b = 20, c = 102

③ a = 101, b = 201, c = 101

④ a = 102, b = 202, c = 102

정답 2번

출제 포인트

이 문제는 전역 변수와 지역 변수의 이름 가림, 정적 지역 변수와 자동 지역 변수의 수명 차이를 이용해 C 프로그램의 실행 결과를 판단하는 문제이다.

핵심 개념

전역 변수: 함수 밖에서 선언되어 프로그램 전체 실행 동안 유지된다.
지역 변수: 함수 안에서 같은 이름으로 선언되면 그 함수 안에서는 전역 변수를 가린다.
정적 지역 변수: 함수 호출이 끝나도 값이 유지되며, 초기화는 한 번만 이루어진다.
자동 지역 변수: 함수가 호출될 때마다 새로 생성되고 초기화되며, 호출이 끝나면 소멸한다.
첫 번째 func() 호출 뒤에는 정적 지역 a가 101, 자동 지역 b가 201, 전역 c가 101이 된다. 두 번째 호출 뒤에는 정적 지역 a가 102, 자동 지역 b가 다시 201, 전역 c가 102가 된다. main()의 printf는 전역 변수 a, b, c를 출력하므로 결과는 a = 10, b = 20, c = 102이다.

NoteBase

선지별 해설

① X — c = 30이 틀린 표현이다. func()에서 c = a가 두 번 실행되어 전역 c는 최종적으로 102가 되므로, 올바르게는 a = 10, b = 20, c = 102이다.
② O — 함수 안의 지역 a와 b가 전역 a와 b를 가리므로 전역 a와 b는 각각 10과 20을 유지한다. 정적 지역 a는 두 번 증가하여 102가 되고 그 값이 전역 c에 저장된다.
③ X — a = 101, b = 201, c = 101이 틀린 표현이다. printf가 출력하는 a와 b는 함수 호출이 끝난 뒤의 전역 변수이므로 각각 10과 20이며, 두 번째 호출까지 반영된 c는 102이다. 올바르게는 a = 10, b = 20, c = 102이다.
④ X — a = 102, b = 202가 틀린 표현이다. 정적 지역 a의 값 102는 전역 a가 아니며, 자동 지역 b는 호출할 때마다 200으로 다시 초기화되어 각 호출에서 201까지만 증가한다. 또한 printf는 전역 a와 b를 출력하므로 올바르게는 a = 10, b = 20, c = 102이다.

오답 포인트

함수 안의 a와 b를 전역 변수와 같은 변수로 보거나, 정적 지역 a의 누적값과 자동 지역 b의 일회성 값을 그대로 printf가 출력한다고 착각하기 쉽다. 특히 두 번째 호출에서도 자동 지역 b는 201이 될 뿐 202로 누적되지 않는다.

다시 나오면 이렇게 판단

먼저 각 이름의 선언 위치로 변수를 구분하고, 같은 이름의 지역 변수가 전역 변수를 가리는지 확인한다. 그다음 static 변수는 호출 사이에 값을 유지하고 일반 지역 변수는 호출마다 초기화된다는 기준으로 호출별 값을 추적한 뒤, 출력문이 어느 범위의 변수를 참조하는지 판단한다.

암기 포인트

지역 변수는 전역 변수를 가리고, static 지역 변수는 누적되며, 일반 지역 변수는 호출마다 다시 시작한다.

NoteBase