2018년 국가직 9급 컴퓨터일반 풀이

by 호이호이꿀떡

정답 체크

01	02	03	04	05	06	07	08	09	10
②	②	①	②	②	③	④	④	③	①
11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
③	①	③	②	③	④	④	③	②	③

문 1. 유닉스 운영체제에 대한 설명으로 옳지 않은 것은?

① 계층적 파일시스템과 다중 사용자를 지원하는 운영체제이다.
② BSD 유닉스의 모든 코드는 어셈블리 언어로 작성되었다.
③ CPU 이용률을 높일 수 있는 다중 프로그래밍 기법을 사용한다.
④ 사용자 프로그램은 시스템 호출을 통해 커널 기능을 사용할 수 있다.

답 ②

② BSD는 1977년부터 1995년까지 빌 조이(Bill Joy)를 주축으로 캘리포니아 대학교 버클리 캠퍼스(University of California, Berkeley)의 CSRG(Computer Systems Research Group)에서 개발한 유닉스 계열의 운영 체제이다.
초기의 유닉스는 어셈블리 언어로 작성되었으나, 같은 연구소의 Dennis Ritchie가 C언어를 개발한 뒤, C언어를 이용하여 유닉스를 다시 만들어, 고급언어로 작성된 최초의 운영체제가 되었다. 따라서 BSD를 포함한 거의 모든 유닉스는 **90%의 C언어와 10%의 어셈블리어로 작성**되어 있다.

<오답 체크> ④ 유닉스의 대부분은 C언어로 작성되어 있지만 커널의 일부분은 효율성 때문에 어셈블리어로 작성되어 있다.
사용자는 직접 커널에 접근할 수는 없고, 시스템 호출(system call) 인터페이스를 통하여 커널을 사용한다.

◈ 유닉스의 특징
▷ 대화식 운영체제
▷ 다중 작업 기능(멀티태스킹)
▷ 다중 사용자 기능
▷ 이식성(하드웨어 종류에 상관없이 운영되는 특성)
▷ 계층적 트리 구조 파일 시스템
▷ 다양한 개발 도구: 여러가지 언어(Fortran, C, C++ 등)를 사용하여 프로그래밍할 수 있도록 많은 컴파일러(compiler)
▷ 여러 가지의 통신 유틸리티 제공
▷ 가상 메모리 사용

▶ 커널(Kernel)
유닉스 시스템의 핵심 부분
메모리 관리, 입출력 장치 관리 등 하드웨어와 관련된 작업을 수행하는 것으로 사용자들은 직접 커널에 접근할 수 없다.

▶ 쉘(shell)
사용자와 시스템 간의 인터페이스를 담당
명령을 해석하는 부분으로서, 사용자가 명령을 입력하면 이를 번역하여 커널에게 넘겨주어 프로그램을 실행하도록 함

문 2. 다음에서 설명하는 해킹 공격 방법은?

> 공격자는 사용자의 합법적 도메인을 탈취하거나 도메인 네임 시스템(DNS) 또는 프락시 서버의 주소를 변조하여, 사용자가 진짜 사이트로 오인하여 접속하도록 유도한 후 개인정보를 훔친다.

① 스니핑(Sniffing)
② 파밍(Pharming)
③ 트로이 목마(Trojan Horse)
④ 하이재킹(Hijacking)

답 ②

② **파밍**(pharming)
사용자가 자신의 웹 브라우저에서 올바른 도메인을 입력해도 가짜 웹 페이지에 접속하게 하여 개인정보를 훔치는 공격
<오답 체크> ① **스니핑**(Sniffing)
네트워크 상에서 자신이 아닌 다른 상대방들의 패킷 교환을 엿듣는 것
③ **트로이목마**(Trojan Horse)
정상적인 프로그램으로 가장한 악성 프로그램으로, 다른 시스템으로 전파되지는 않는다. 트로이목마는 보통 해커들이 대상 컴퓨터의 인증이나 백신을 우회하여 시스템 내부에 침투하기 위해 사용한다.
④ **세션 하이재킹**(Session Hijacking) 공격
시스템에 접근할 적법한 사용자 아이디와 패스워드를 모를 때, 이미 시스템에 접속되어 세션이 연결되어 있는 사용자의 세션을 가로채는 공격이다.

문 3. 다음 SQL 명령어에서 DDL(Data Definition Language) 명령어만을 모두 고른 것은?

> ㄱ. ALTER
> ㄴ. DROP
> ㄷ. INSERT
> ㄹ. UPDATE

① ㄱ, ㄴ
② ㄴ, ㄷ
③ ㄴ, ㄹ
④ ㄷ, ㄹ

답 ①

① ㄱ. ㄴ. DDL(데이터 정의어)
<오답 체크> ㄷ. ㄹ. DML(데이터 조작어)

▶ 데이터 정의어(DDL, Data Definition Language)
- CREATE: 스키마, 테이블, 뷰 등 데이터베이스 객체 만들기
- DROP: 데이터베이스 객체 삭제
- ALTER: 데이터베이스 객체 변경
- TRUNCATE: 테이블에서 모든 행 빠른 삭제, 복구불가

▶ 데이터 조작어(DML, Data Manipulation Language)
- SELECT: 데이터 검색(질의)
- INSERT: 데이터 삽입(등록)
- UPDATE: 데이터 업데이트 (수정)
- DELETE: 데이터 삭제

▶ 데이터 제어어(DCL, Data Control Language)
- GRANT: 권한을 부여
- REVOKE: 권한을 박탈
- SET TRANSACTION: 트랜잭션 모드 설정
- BEGIN: 트랜잭션 시작
- COMMIT: 트랜잭션 실행
- ROLLBACK: 트랜잭션 실행 취소
- SAVEPOINT: 롤백 지점 설정
- LOCK: 데이터 자원을 차지

문 4. 다음 수식에서 이진수 Y의 값은?(단, 수식의 모든 수는 8 비트 이진수이고 1의 보수로 표현된다)

$$11110100_{(2)} + Y = 11011111_{(2)}$$

① $11101001_{(2)}$
② $11101010_{(2)}$
③ $11101011_{(2)}$
④ $11101100_{(2)}$

답 ②

② 1의 보수에서 양수와 음수는 각각의 비트를 반전시키기만 하면 된다.
그러므로 11110100는 00001011의 음수값이 된다.
00001011은 십진수 11이므로, 11110100은 십진수 -11
마찬가지로 11011111은 십진수 -32

따라서 위의 식은 -11 + Y = -32이며, Y = -21이 된다.
21은 이진수로 00010101이므로, **Y = -21은 11101010**이다.

다른 풀이) 식에서 Y를 제외한 11110100을 우측으로 넘긴다.
Y = 11011111 - 11110100
1의 보수에서 양수와 음수는 비트값들을 반전시키기만 하면 되기 때문에, -11110100은 +00001011이 된다.
Y = 11011111 + 00001011 = **11101010**

문 5. 다음 진리표를 만족하는 부울 함수로 옳은 것은? (단, •은 AND, ⊕는 XOR, ⊙는 XNOR 연산을 의미한다)

입력			출력
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

① Y = A • B ⊕ C
② Y = A ⊕ B ⊙ C
③ Y = A ⊕ B ⊕ C
④ Y = A ⊙ B ⊙ C

답 ②

먼저 진리표에서 결과가 참(1)이 나오는 경우를 추출하여 불 대수로 정리해야 한다.
선지를 보면 알듯이 이걸 풀기 위해선 XOR(⊕)과 XNOR(⊙)의 불 대수를 알아야 한다.
먼저 XOR(⊕)은 두 입력값이 다를 경우 true(1)을 출력하기 때문에 **Y = A ⊕ B = AB' + A'B** 이다.
XNOR(⊙)는 두 입력값이 같을 경우 true(1)을 출력하기 때문에 **Y = A ⊙ B = AB + A'B'** 이다.
또한 XOR과 XNOR이 반대인 것을 알 수 있다.
(A ⊕ B)' = A ⊙ B
진리표에서 결과가 true(1)인 경우는 A'B'C', A'BC, AB'C, ABC' 이므로,
Y = A'B'C' + A'BC + AB'C + ABC'
= A' (B'C' + BC) + A (B'C + BC')
= A' (B ⊙ C) + A (B ⊕ C) < ------- (ㄱ)
= A' (B ⊙ C) + A (B ⊙ C)'
= A ⊕ (B ⊙ C) = **A ⊕ B ⊙ C**

또한 (ㄱ)에서
A' (B ⊙ C) + A (B ⊕ C)
= A' (B ⊕ C)' + A (B ⊕ C)
= A ⊙ (B ⊕ C) = **A ⊙ B ⊕ C**

이를 통해 ⊕와 ⊙는 교환법칙이 성립하는 것을 알 수 있다.

문 6. 스레싱(Thrashing)에 대한 설명으로 옳지 않은 것은?

① 프로세스의 작업 집합(Working Set)이 새로운 작업 집합으로 전이 시 페이지 부재율이 높아질 수 있다.
② 작업 집합 기법과 페이지 부재 빈도(Page Fault Frequency) 기법은 한 프로세스를 중단(Suspend) 시킴으로써 다른 프로세스들의 스레싱을 감소시킬 수 있다.
③ 각 프로세스에 설정된 작업 집합 크기와 페이지 프레임 수가 매우 큰 경우 다중 프로그래밍 정도(Degree of Multiprogramming)를 증가시킨다.
④ 페이지 부재 빈도 기법은 프로세스의 할당받은 현재 페이지 프레임 수가 설정한 페이지 부재율의 하한보다 낮아지면 보유한 프레임 수를 감소시킨다.

답 ③

③ 페이지 프레임을 프로세스들에게 분배하는 방법에는 균등 할당법과 비례 할당법이 있다.
▷ 균등 할당법
모든 프로세스에게 똑같은 수의 프레임을 배정하는 방법이다.
▷ 비례 할당법
각 프로세스마다 다르게 할당하는 방법으로, 각 프로세스는 필요로 하는 양에 비례하여 할당 받는다.
주기억장치의 크기는 한정되어 있는데 하나의 프로세스가 차지하는 작업 집합이 크거나 할당된 페이지 프레임 수가 많은 경우, 동시에 실행할 수 있는 프로세스의 수는 당연히 적어질 것이다. 따라서 다중 프로그래밍 정도는 낮아진다.
<오답 체크> ① 기존에 페이지 프레임에 들어있던 프로그램 및 데이터들은 기존 작업 집합을 위한 것이기 때문에, 새로운 작업 집합을 하기 위해선 새로운 프로그램과 데이터를 페이지 프레임으로 불러와야 하기 때문에 페이지 부재율이 높아진다.
② **스래싱**(thrashing)
다중 프로그래밍 정도를 높이게 되면 주기억장치에 없는 데이터를 요구하는 횟수가 계속 증가하게 되고, 나중엔 프로세스가 데이터를 처리하는 시간보다 페이지 교체에 더 많은 시간을 허비하게 되어 결국 CPU 이용률이 현저히 떨어지게 되는 현상
당장 이용하지 않을 프로세스를 중단 시킴으로써 스래싱을 감소시킬 수 있다.
④ 페이지 프레임 비례 할당법은 각 프로세스가 필요로 하는 양에 비례하여 페이지 프레임 수를 조절한다. 현재 프로세스의 페이지 부재율이 하한보다 낮아지면, 보유한 페이지 프레임 수를 줄여 다른 프레로세스에게 양보한다.

문 7. 인공신경망에 대한 설명으로 옳은 것만을 모두 고른 것은?

> ㄱ. 단층 퍼셉트론은 배타적 합(Exclusive-OR) 연산자를 학습할 수 있다.
> ㄴ. 다층 신경망은 입력 층, 출력 층, 하나 이상의 은닉 층들로 구성된다.
> ㄷ. 뉴런 간 연결 가중치(Connection Weight)를 조정하여 학습한다.
> ㄹ. 생물학적 뉴런 망을 모델링한 방식이다.

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

답 ④

ㄴ. 다층 신경망(Multi-Layer neural Network)은 다층 퍼셉트론을 이용한 인공신경망을 말하며, 다층 퍼셉트론은 입력층, 출력층, 하나 이상의 은닉층들로 구성된다.
단층 퍼셉트론은 은닉층 없이 입력층과 출력층으로만 구성된다.

ㄷ. 퍼셉트론의 각 노드의 가중치와 입력치를 곱한 것을 모두 합한 값(연결 가중치)이 활성함수에 의해 판단되는데, 그 값이 임계치(보통 0)보다 크면 뉴런이 활성화되고 결과값으로 1을 출력한다. 뉴런이 활성화되지 않으면 결과값으로 -1을 출력한다.

ㄹ. 생물학적 뉴런 망을 모델링한 방식이다.

<오답 체크> ㄱ. 단층 퍼셉트론은 비선형함수를 학습할 수 없으며, 은닉층이 없어 XOR(배타적 합) 연산을 할 수 없다.

◈ **인공신경망**(ANN, Artificial Neural Network)
생물학의 신경망에서 영감을 얻은 통계학적 학습 알고리즘으로, 시냅스의 결합으로 네트워크를 형성한 인공 뉴런(노드)이 학습을 통해 시냅스의 결합 세기를 변화시켜 문제 해결 능력을 가지는 기계학습 모델이다.

▷ **퍼셉트론**(Perceptron)은 뉴런의 수학적 모델을 일컫는 용어이기도 하고, 최초로 제안된 신경망 프로그램 알고리즘이기도 하다.
퍼셉트론은 선형함수를 이용해 하나의 뉴런을 사용하며 학습 데이터를 가장 잘 설명할 수 있는 최적의 패러미터 찾는다.
각 노드의 가중치와 입력치를 곱한 것을 모두 합한 값이 활성함수에 의해 판단되는데, 그 값이 임계치(보통 0)보다 크면 뉴런이 활성화되고 결과값으로 1을 출력한다. 뉴런이 활성화되지 않으면 결과값으로 -1을 출력한다.
그러나 (단층) 퍼셉트론은 비선형함수를 학습할 수 없으며, 입력층과 출력층만 있기에 XOR 문제도 해결할 수 없다.

▷ **다층 퍼셉트론**(**MLP,** Multi Layer Perceptron)
여러 개의 퍼셉트론을 연결시켜 층(Layer)을 만들고, 이 층들을 중첩시켜 다층(Multi Layer)으로 만든 것으로, 단층 퍼셉트론에서 문제가 되었던 XOR 연산이 가능하다고 한다.
입력층(Input layer), 은닉층(Hidden layer), 출력층(Output layer)으로 구분되어 있으며, 필요이상으로 많은 층을 두는 것은 오히려 성능이 떨어진다고 알려져 있다.

▷ 입력층(Input Layer): 외부로부터 입력자료를 받아들여 시스템으로 전달

▷ 은닉층(Hidden Layer): 시스템 안쪽에 위치하고 있으며, 입력값을 넘겨받아 처리한 뒤 결과를 산출

▷ 출력층(Output Layer): 입력값과 시스템 상태를 기준하여 시스템 출력값을 산출

문 8. 네트워크 기술에 대한 설명으로 옳지 않은 것은?

① IPv6는 인터넷 주소 크기가 128비트이고 호스트 자동 설정기능을 제공한다.
② 광대역통합망은 응용 서비스별로 약속된 서비스 레벨 보증(Service Level Agreement) 품질 수준을 보장해줄 수 있다.
③ 모바일 와이맥스(WiMAX)는 휴대형 단말기를 이용해 고속 인터넷 접속 서비스를 제공하는 무선망 기술이다.
④ SMTP(Simple Mail Transfer Protocol)는 사용자 인터페이스 구성방법을 지정하는 전송 계층 프로토콜이다.

답 ④

④ SMTP는 전자메일 전송을 위한 응용 계층 프로토콜이다.
전송 계층 프로토콜로는 TCP나 UDP가 있다.

<오답 체크> ① IPv6는 16비트씩 8자리, 총 128비트(16바이트)
호스트 주소 자동 설정 기능: IPv6 호스트는 IPv6 네트워크에 접속하는 순간 자동적으로 네트워크 주소를 부여 받는다.
IPv4는 네트워크 관리자로부터 IP 주소를 부여받아 수동으로 설정해야 한다.

② **광대역통합망**(BCN, Broadband Convergence Network)
통신, 방송, 인터넷이 융합된 품질 보장형, 광대역 멀티미디어 서비스를 언제 어디서나 끊김없이 안전하게 광대역으로 이용할 수 있는 차세대 통합 네트워크

▷ 통합 네트워크에서의 다양한 서비스 제공
▷ 표준화된 개방형 네트워크 구조
▷ 패킷 기반의 통합형 네트워크
▷ 품질 보장형 광대역 서비스
▷ 일반화된 이동성(Mobility) 제공

③ **모바일 와이맥스**(Mobile WiMAX, 와이브로(WiBro))
한국전자통신연구원과 삼성전자가 개발한 무선 광대역 인터넷 기술로, 휴대전화처럼 언제 어디서나 이동하면서 초고속인터넷을 이용할 수 있는 서비스이다. 국제 표준은 IEEE 802.16e 이다.

문 9. 다음 Java 프로그램의 출력 값은?

```
class Super {
    Super() {
        System.out.print('A');
    }

    Super(char x) {
    System.out.print(x);
    }
}

class Sub extends Super {
    Sub() {
        super();
        System.out.print('B');
    }

    Sub(char x) {
        this();
        System.out.print(x);
    }
}

public class Test {
    public static void main(String[] args) {
        Super s1 = new Super('C');
        Super s2 = new Sub('D');
    }
}
```

① ABCD
② ACBD
③ CABD
④ CBAD

답 ③

Super 클래스는 부모 클래스(상위 클래스)
Sub 클래스는 자식 클래스(하위 클래스)

Super 클래스와 Sub 클래스에는 각각 같은 이름의 다른 매개변수를 가진 메소드들이 있다. 이것을 **오버로딩**(Overloading)이라 하며, 메소드를 사용할 때 변수 타입을 확인하여야 한다.

Super s1 = new Super('C');에서 Super 객체인 s1을 생성하고, Super('C')에 의해 선언과 동시에 메소드가 실행된다.
Super('C')이므로 Super 클래스의 Super(char x) 메소드가 실행, **'C'가 출력**된다.

Super s2 = new Sub('D');에서 부모인 Super 클래스를 참조하는 Sub 타입 객체 s2를 생성하고, Sub('D')에 의해 선언과 동시에 메소드를 실행한다.
Sub('D')이므로 Sub 클래스의 Sub(char x) 메소드가 실행된다.

Sub(char x) 메소드의 첫 줄은 this() 메소드인데, this()는 자기 자신의 생성자를 호출함으로써 생성자의 초기화 과정을 생략할 수 있게 해주는 메소드이다.
this()에 의해 자기 자신의 생성자인 Sub가 매개변수 없이 Sub()로 호출된다.

그런데 이 때 Sub() 메소드 안에서 super() 메소드를 만나게 되는데, super()는 상속받은 바로 위 클래스의 생성자를 호출하는메소드이다.
super()에 의해 상위 클래스인 Super가 매개변수 없이 Super()로 호출되어, **'A'를 출력**한다.
(여기서 사용자가 정의한 대문자 Super() 메소드와 소문자 super()메소드는 다르다. 소문자 super() 메소드는 자바에서 미리 정의되어 있는 키워드이다.)

이후 Sub() 메소드로 돌아와 super(); 아래인
System.out.print('B'); 행이 실행되어, **'B'가 출력**된다.

그리고 Sub(char x) 메소드로 돌아와 this(); 아래인
System.out.print(x); 행이 실행되어, 처음에 생성하면서 넘겨받은 변수값 **'D'가 출력**된다.

문 10. 개발자가 사용해야 하는 서브시스템의 가장 앞쪽에 위치하면서 서브시스템에 있는 객체들을 사용할 수 있도록 인터페이스 역할을 하는 디자인 패턴은?

① Facade 패턴
② Strategy 패턴
③ Adapter 패턴
④ Singleton 패턴

답 ①

◈ 디자인 패턴(Design pattern)
객체 지향 프로그래밍 설계를 할 때 자주 발생하는 문제들을 피하기 위해 사용되는 패턴

▶ 안티패턴(anti-pattern)
실제 많이 사용되는 패턴이지만 비효율적이거나 비생산적인 패턴을 의미

① Facade pattern(퍼사드 패턴)
어떤 서브시스템의 일련의 인터페이스에 대한 통합된 인터페이스를 제공한다. 고수준 인터페이스를 정의하기 때문에 서브시스템을 더 쉽게 사용할 수 있도록 해준다.

<오답 체크> ② Strategy pattern(스트래티지 패턴)
알고리즘군을 정의하고 각각 캡슐화하여 교환해서 사용할 수 있도록 만든다. 스트래티지 패턴을 활용하면 알고리즘을 사용하는 클라이언트와는 독립적으로 알고리즘을 변경할 수 있다.

③ Adapter pattern(어댑터 패턴)
한 클래스의 인터페이스를 클라이언트에서 사용하고자 하는 다른 인터페이스로 변환한다. 어댑터를 이용하면 인터페이스 호환성 문제 때문에 같이 쓸 수 없는 클래스들을 연결해서 쓸 수 있다.

④ Singleton pattern(싱글턴 패턴)
해당 클래스의 인스턴스가 하나만 만들어지고, 어디서든지 그 인스턴스에 접근할 수 있도록 하기 위한 패턴

이 외 디자인 패턴 몇 가지

▶ Observer pattern (옵저버 패턴)
한 객체의 상태가 바뀌면 그 객체에 의존하는 다른 객체들한테 알림으로써 자동으로 내용이 갱신되는 방식으로, 일대다(one-to-many) 의존성을 정의한다.

▶ Decorator pattern(데코레이터 패턴)
객체에 추가적인 요건을 동적으로 첨가한다. 데코레이터는 서브클래스를 만드는 것을 통해서 기능을 유연하게 확장할 수 있는 방법을 제공한다.

▶ Factory method pattern(팩토리 메소드 패턴)
객체를 생성하기 위한 인터페이스를 정의를 서브클래스에서 결정하게 만든다. 즉 팩토리 메소드 패턴을 이용하면 클래스의 인스턴스를 만드는 일을 서브클래스에게 맡는다.

▶ Abstract Factory Pattern(추상 팩토리 패턴)
인터페이스를 이용하여 서로 연관된, 또는 의존하는 객체를 구상 클래스를 지정하지 않고도 생성

▶ Command pattern(커맨드 패턴)
요구사항을 객체로 캡슐화 할 수 있으며, 매개변수를 써서 여러 가지 다른 요구 사항을 집어넣을 수 있다. 또한 요청 내역을 큐에 저장하거나 로그로 기록할수 도있으며 작업취소 기능도 지원 가능 하다.

▶ Template method pattern(템플릿 메소드 패턴)
메소드에서 알고리즘의 골격을 정의한다. 알고리즘의 여러 단계 중 일부는 서브클래스에서 구현할 수 있다. 템플릿 메소드를 이용하면 알고리즘의 구조는 그대로 유지하면서 서브클래스에서 특정 단계를 재정의 할 수 있다.

▶ Iterator pattern(이터레이터 패턴)
컬렉션 구현 방법을 노출시키지 않고도 그 집합체 안에 들어있는 모든 항목에 접근할 수 있는 방법을 제공한다.

▶ Composite pattern(컴포지트 패턴)
객체들을 트리 구조로 구성하여 부분과 전체를 나타내는 계층구조로 만들 수 있다. 클라이언트에서 개별 객체와 다른 객체들로 구성된 복합 객체(composite)를 똑같은 방법으로 다룰 수 있다.

문 11. 소프트웨어 모듈 평가 기준으로 판단할 때, 다음 4 명 중 가장 좋게 설계한 사람과 가장 좋지 않게 설계한 사람을 순서대로 바르게 나열한 것은?

○ 철수: 절차적 응집도 + 공통 결합도
○ 영희: 우연적 응집도 + 내용 결합도
○ 동수: 기능적 응집도 + 자료 결합도
○ 민희: 논리적 응집도 + 스탬프 결합도

① 철수, 영희
② 철수, 민희
③ 동수, 영희
④ 동수, 민희

답 ③

▷ 응집도(cohesion)는 모듈 내부의 기능들이 관련되어 있는 정도, 결합도(coupling)는 모듈과 모듈 간의 상호 의존 정도를 나타낸 것이다. 응집도는 높을수록, 결합도는 낮을수록 좋다.

③ 응집도가 가장 높은 것은 기능적 응집도이며, 가장 낮은 것은 우연적 응집도이다.
또한 결합도가 가장 낮은 것은 자료 결합도(데이터 결합도)이며, 가장 높은 것은 내용 결합도이다.
따라서 동수의 설계가 가장 좋고, 영희도 설계가 가장 안 좋다.

◈ 응집도(cohesion) 높은(좋은) 순서대로
▶ 기능적 응집도(Functional Cohesion): 모듈 내부의 모든 기능 요소들이 단일한 목적을 위해 수행되는 경우
▶ 순차적 응집도(Sequential Cohesion): 모듈 내에서 한 기능 요소로부터 나온 출력값을 다른 기능에서 사용할 경우
▶ 교환적 응집도(Communication Cohesion): 동일한 입력과 출력을 사용하여 다른 기능을 수행하는 활동들이 모여있을 경우
▶ 절차적 응집도(Procedural Cohesion): 모듈이 다수의 관련 기능을 가질 때 모듈 안의 구성요소들이 그 기능을 순차적으로 수행할 경우
▶ 시간적 응집도(Temporal Cohesion): 연관된 기능이라기보다 특정 시간에 처리되어야 하는 활동들을 한 모듈에서 처리할 경우
▶ 논리적 응집도(Logical Cohesion): 유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들이 한 모듈에서 처리되는 경우
▶ 우연적 응집도(Coincidental Cohesion): 모듈 내부의 각 구성요소들이 연관이 없는 경우

◈ 결합도(coupling) 낮은(좋은) 순서대로
▶ 자료 결합도(Data Coupling): 모듈간의 인터페이스가 자료 요소로만 구성된 경우. 파라미터를 통해서만 모듈간의 상호 작용이 일어나는 경우. Call by value
▶ 스탬프 결합도(Stamp Coupling): 모듈간의 인터페이스로 배열이나 오브젝트, 스트럭쳐등이 전달되는 경우
▶ 제어 결합도(Control Coupling): 단순히 처리를 해야 할 대상인 값만 전달되는 게 아니라 어떻게 처리를 해야 한다는 제어 요소(DCD, Flag등)이 전달되는 경우.
▶ 외부 결합도(External Coupling): 어떤 모듈에서 반환한 값을 다른 모듈에서 참조해서 사용하는 경우
▶ 공통 결합도(Common Coupling): 파라미터가 아닌 모듈 밖에 선언되어 있는 전역 변수를 참조하고 전역변수를 갱신하는 식으로 상호작용하는 경우
▶ 내용 결합도(Content Coupling): 다른 모듈 내부에 있는 변수나 기능을 다른 모듈에서 사용 하는 경우

문 12. 자료구조에 대한 설명으로 옳지 않은 것은?

① 데크는 삽입과 삭제를 한쪽 끝에서만 수행한다.
② 연결리스트로 구현된 스택은 그 크기가 가변적이다.
③ 배열로 구현된 스택은 구현이 간단하지만 그 크기가 고정적이다.
④ 원형연결리스트는 한 노드에서 다른 모든 노드로 접근이 가능하다.

답 ①

① **데크**(Deque, Double Ended Queue)
양쪽 끝에서 삽입과 삭제가 가능한 선형 리스트
▷ 입력 제한 데크(Scroll): 입력을 한쪽 끝에서만 가능하게 한 데크
▷ 출력 제한 데크(Shelf): 출력을 한쪽 끝에서만 가능하게 한 데크
<오답 체크> ② 연결 리스트(linked list)는 크기를 변경시키는 게 쉽게 때문에 스택의 크기 역시 변경할 수 있다.
③ 배열(array)은 선언할 때 크기를 지정하기 때문에 스택의 크기가 고정적이다.
④ 원형 연결 리스트는 단일 연결 리스트와는 다르게 마지막 노드의 포인터가 다시 첫 노드의 주소를 가리키는 구조로, 마지막 노드와 첫 노드가 연결된 리스트이다. 따라서 한 노드에서 다른 모든 노드로 접근이 가능하다.

문 13. IPv4 가 제공하는 기능만을 모두 고른 것은?

ㄱ. 혼잡제어
ㄴ. 인터넷 주소지정과 라우팅
ㄷ. 신뢰성 있는 전달 서비스
ㄹ. 패킷 단편화와 재조립

① ㄱ, ㄴ
② ㄴ, ㄷ
③ ㄴ, ㄹ
④ ㄷ, ㄹ

답 ③

ㄴ. ㄹ. IP는 OSI 3계층 네트워크 계층 프로토콜로, 비연결형이며 신뢰성을 보장하지 않는다. 인터넷 경로설정을 위한 라우팅 역할을 수행하며, 패킷의 분해(단편화), 조립, 논리적 경로지정 기능을 제공한다.
순서 검사(sequence checking)와 흐름 제어, 오류 복구 기능 등이 없다.
<오답 체크> ㄱ. ㄷ. OSI 4계층인 전송 계층의 연결지향형 프로토콜인 TCP에 대한 설명이다.

문 14. 결정 명령문 내의 각 조건식이 참, 거짓을 한 번 이상 갖도록 조합하여 테스트 케이스를 설계하는 방법은?

① 문장 검증 기준(Statement Coverage)
② 조건 검증 기준(Condition Coverage)
③ 분기 검증 기준(Branch Coverage)
④ 다중 조건 검증 기준(Multiple Condition Coverage)

답 ②

② **조건 검증 기준**(Condition Coverage): 모든 분기점에서 조건식을 구성하는 단일 조건의 참과 거짓을 한 번 이상 실행
<오답 체크> ① **문장 검증 기준**(Statement Coverage): 프로그램의 모든 문장을 한 번 이상 실행
③ **분기 검증 기준**(Branch Coverage): 모든 분기점에서 참과 거짓에 해당하는 경로를 한 번 이상 실행
④ **복수 조건 검증 기준**(Multiple Condition Coverage): 조건식을 구성하는 단일 조건식들의 모든 가능한 참/거짓 조합을 한 번 이상 실행
▶ **조건/분기 검증 기준**(modified condition/decision coverage): 조건 검증 기준과 분기 검증 기준을 모두 만족해야 함
▶ **경로 검증 기준**(Path Coverage): 프로그램에 존재하는 모든 실행 가능한 경로를 한 번 이상 테스트함. 반복 문장이 있다면 실행 가능한 경로의 수는 무한대이므로 사실상 불가능
▶ **기본 경로 테스트**(Basic Path Testing): 시작 노드에서 종료 노드까지의 선형 독립적인 경로(기본 경로)를 모두 테스트 메케이브의 사이클로매틱 수는 기본 경로의 개수와 일치함

문 15. 가상 머신(Virtual Machine)에 대한 설명으로 옳지 않은 것은?

① 단일 컴퓨터에서 가상화를 사용하여 다수의 게스트 운영체제를 실행할 수 있다.
② 가상 머신은 사용자에게 다른 가상 머신의 동작에 간섭을 주지 않는 격리된 실행환경을 제공한다.
③ 가상 머신 모니터(Virtual Machine Monitor)를 사용하여 가상화하는 경우 반드시 호스트 운영체제가 필요하다.
④ 자바 가상 머신은 자바 바이트 코드가 다양한 운영체제 상에서 수행될 수 있도록 한다.

답 ③

◈ **가상 머신**(Virtual Machine, VM)
컴퓨팅 환경을 소프트웨어로 구현한 것, 즉 컴퓨터를 에뮬레이션하는 소프트웨어이다. 가상 머신 상에서 운영 체제나 응용 프로그램을 설치 및 실행할 수 있다.
③ 하드웨어에 직접 가상 머신 모니터(하이퍼바이저)를 설치하여 사용하여 네이티브 베어메탈 방식의 가상화의 경우 별도의 호스트 운영체제가 필요없다.

◈ **가상화 머신 모니터**(virtual machine monitor, **VMM**)는 **하이퍼바이저**(hypervisor)라고도 하며, 물리적인 리소스로부터 운영체제와 애플리케이션을 분리해주는 가상화(Virtualization)를 구현하기 위한 논리적 플랫폼(platform)을 의미한다.
▶ 타입1 하이퍼바이저: **네이티브 베어메탈**(Native(bare-metal)) 방식
하이퍼바이저와 OS의 합본
물리적 컴퓨터(하드웨어)에 하이퍼바이저를 바로 설치하는 구조
하이퍼바이저에 OS 기능(하드웨어 제어 등)이 포함되어, 별도의 호스트 OS가 없다.
하이퍼바이저가 하드웨어에 직접 설치된 구조라서, 호스트형 가상화에 비해 오버헤드가 적고 물리적 컴퓨터 리소스 관리가 유연하다.
반면, 자체적으로 관리기능을 갖고 있지 않아 별도의 관리 콘솔 혹은 관리 컴퓨터가 필요하다는 단점이 있다.
▶ 타입2 하이퍼바이저 **호스트형**(Hosted) 방식
OS 설치 후 별도로 하이퍼바이저 설치
물리적 컴퓨터에 일반 OS를 설치하고 하이퍼바이저 소프트웨어를 설치하는 구조
물리 컴퓨터의 하드웨어를 소프트웨어적으로 구현하여 에뮬레이트하는 방식으로, 네이티브 방식에 비해 오버헤드가 크다.
하지만 게스트 OS 의 종류에 제약이 적고, 손쉽게 도입이 가능하며, 한 컴퓨터에 다른 종류의 하이퍼바이저를 2가지 이상 설치할 수 있다는 장점이 있다.

문 16. IEEE 802.11 무선 랜에 대한 설명으로 옳은 것은?

① IEEE 802.11a 는 5 GHz 대역에서 5.5 Mbps 의 전송률을 제공한다.
② IEEE 802.11b 는 직교 주파수 분할 다중화(OFDM) 방식을 사용하여 최대 22 Mbps 의 전송률을 제공한다.
③ IEEE 802.11g 는 5 GHz 대역에서 직접 순서 확산 대역(DSSS) 방식을 사용한다.
④ IEEE 802.11n 은 다중입력 다중출력(MIMO) 안테나 기술을 사용한다.

답 ④

◈ 802.11: 무선 인터넷을 위한 일련의 표준 규격
▶ 802.11(초기 버전)
최고속도 2Mbps / 2.4GHz 대역 / CSMA/CA 기술
▶ **802.11a**
최고 54Mbps 속도 / 5GHz 대역 / OFDM 기술
▶ **802.11b**
최고 전송속도 11Mbps이나 실제로는 6-7Mbps 정도의 효율 / 2.4GHz 대역 / HR-DSSS 기술
▶ **802.11g**
최고 24또는 54Mbps / 2.4GHz 대역 / OFDM, DSSS 기술
널리 사용되고 있는 802.11b 규격과 쉽게 호환
▶ **802.11n**
최고 600Mbps / 2.4GHz과 5GHz 대역 사용
MIMO(multiple-input multiple-output)와 40 MHz 채널 대역폭을 가진 물리 계층(physical layer), 맥 계층(MAC layer)의 프레임 집적(frame aggregation) 기술
▶ 802.11ac
다중 단말의 무선랜 속도는 최소 1 Gbit/s, 단일 링크 속도는 최소 500 Mbit/s 까지 가능
더 넓은 무선 주파수 대역폭(최대 160 MHz)
더 많은 MIMO 공간적 스트림(최대 8 개), 다중 사용자 MIMO, 그리고 높은 밀도의 변조(최대 256 QAM)
▶ 802.11ad
빔포밍 기술을 이용하여 최대 7Gb/s의 속도 / 60GHz 대역
대용량의 데이터나 무압축 HD 비디오등 높은 비트레이트 동영상 스트리밍에 적합
하지만 60GHz는 장애물 통과가 어려워 10m이내 같은 공간 내에서만 사용이 가능하여 근거리 사용 기기만 이용 가능
기존 2.4/5GHz 대역 사이도 원활한 전환을 위해 '빠른 세션 전송'을 추가했으며 Tri-band 네트워킹, 무선 도킹, 유선과 동등한 데이터 전송속도, 압축 스트리밍 비디오 지원 등의 보완

문 17. 데이터베이스의 동시성 제어에 대한 설명으로 옳지 않은 것은? (단, T1, T2, T3 는 트랜잭션이고, A 는 데이터 항목이다)

① 다중버전 동시성 제어 기법은 한 데이터 항목이 변경될 때 그 항목의 이전 값을 보존한다.

② T1 이 A 에 배타 로크를 요청할 때, 현재 T2 가 A 에 대한 공유 로크를 보유하고 있고 T3 가 A 에 공유 로크를 동시에 요청한다면, 트랜잭션 기아 회피기법이 없는 경우 A 에 대한 로크를 T3 가 T1 보다 먼저 보유한다.

③ 로크 전환이 가능한 상태에서 T1 이 A 에 대한 배타 로크를 요청할 때, 현재 T1 이 A 에 대한 공유 로크를 보유하고 있는 유일한 트랜잭션인 경우 T1 은 A 에 대한 로크를 배타 로크로 상승할 수 있다.

④ 2 단계 로킹 프로토콜에서 각 트랜잭션이 정상적으로 커밋될 때까지 자신이 가진 모든 배타적 로크들을 해제하지 않는다면 모든 교착상태를 방지할 수 있다.

답 ④

④ **교착상태**(deadlock)는 두 개 이상의 작업이 서로 상대방의 작업이 끝나기만을 기다리고 있기 때문에 무한 대기상태가 되어있는 상태를 말한다. 각 트랜잭션이 각자 점유하고 있는 락을 해제하지 않고 계속 가지고 있는 것이 교착상태의 발생 이유이다.

<오답 체크> ② 공유락은 상호 허용이 되지만, 배타락은 허용이 되지 않는다. 따라서, 현재 T2가 공유락으로 보유중인 A를 T3는 요청하고 접근할 수 있지만, T1은 아직 접근할 수 없다.

③ 현재 A에 대해 접근중인 트랙잭션은 T1하나뿐이기 때문에, 배타락으로 상승(확장)이 가능하다.

▷ 확장단계: 트랜잭션이 필요한 락을 획득하는 단계로, 이 단계에서는 이미 획득한 락을 해제하지 않는다.

▷ 수축단계: 트랜잭션이 락을 해제하는 단계로, 이 단계에서는 새로운 락을 획득하지 않는다.

◈ 동시성 제어(Currency Control)

둘 이상의 트랜잭션이 동시에 수행될 때, 일관성을 해치지 않도록 트랜잭션의 데이터 접근 제어

▶ 기본 락킹 기법

트랜잭션이 데이터에 접근하기 전에 먼저 락킹 연산 수행

언제나 어떤 한 데이터 항목에 대해 하나의 트랜잭션만이 락킹 가능

▶ 2단계 락킹 규약(Twoo-Phase Locking Protocol)

모든 트랜잭션들이 Lock과 Unlock 연산을 2단계로 구분하여 실행

▶ 타입 스탬프(Timestamp) 기법

각 트랜잭션이 시스템에 들어오는 순서대로 타임스탬프를 부여하여 트랜잭션간의 순서를 미리 지정

상충되는 연산들을 타임스탬프 순서대로 처리함으로써 직렬성을 보장

▶ 낙관적 기법

트랜잭션 수행 동안은 어떠한 검사도 하지 않고, 트랜잭션 종료시에 일괄적으로 검사

연산은 버퍼에서 이루어지며, 직렬 가능성 위반성 여부를 검증한 후 결과를 디스크에 반영하거나 복구

읽기 연산이 많은 경우에 적합하며, 데드락과 연쇄 복구가 발생하지 않음

▶ 다중 버전 기법

데이터 항목이 변경될 때 그 데이터 항목의 이전 값을 보존하는 기법

한 데이터 항목에 대해 여러 개의 버전을 유지, 많은 저장 공간이 필요

다수의 트랜잭션들이 실행 중 특정 트랜잭션의 처리를 복구해야 하는 경우, 다른 트랜잭션이 처리한 부분에 대해서도 연쇄적으로 복구해야 하는 현상 초래

문 18. 파일구조에 대한 설명으로 옳지 않은 것은?

① VSAM 은 B+ 트리 인덱스 구조를 사용한다.
② 히프 파일은 레코드들을 키 순서와 관계없이 저장할 수 있다.
③ ISAM 은 레코드 삽입을 위한 별도의 오버플로우 영역을 필요로 하지 않는다.
④ 순차 파일에서 일부 레코드들이 키 순서와 다르게 저장된 경우, 파일 재구성 과정을 통해 키 순서대로 저장될 수 있다.

답 ③

③ **ISAM**(Indexed Sequential Access Method, 색인 순차 파일)
인덱스을 통한 랜덤 처리와 순차 처리를 병행할 수 있는 파일
▷ 인덱스 구역(index area): 기본 데이터 영역에 대한 색인을 구성
▷ 기본 데이터 구역(prime data area): 실제 데이터가 기록되는 부분
▷ 오버플로우 구역(overflow area): 기본 데이터 구역에 기록되지 못하고 넘친 데이터를 기록하는 부분

<오답 체크> ① **VSAM**(Virtual Storage Access Method, 가상 기억 접근 방식)
IBM에서 B+ 트리 구조를 기반으로 개발한 접근 방식으로, 가상 기억 환경에서 직접 접근과 순차 접근의 2가지 방식으로 기억 장치에 있는 데이터를 읽거나 기록할 수 있게 하는 접근 방식을 가능하게 한다.

② **힙파일**은 가장 단순한 파일 조직으로 레코드들이 삽입된 순서대로 파일에 저장된다.(비순서파일)
레코드들 간의 순서를 따지지 않고 파일의 가장 끝에 첨부되기 때문에 삽입 시 성능이 좋다. 삭제 시에도 레코드를 찾아 삭제하고 빈 공간은 그대로 놔둔다. 삭제된 빈 공간을 재사용하기 위해서는 빈 공간을 찾아야 하며, 삭제된 레코드들로 인한 단편들이 발생하여 성능이 저하된다는 단점이 있다.

④ **순차 파일**(Sequential File)
입력되는 데이터의 논리적인 순서에 따라 물리적으로 연속된 위치에 기록한 파일

문 19. 다음 C 프로그램의 출력 값은?

```
#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);

    return 0;
}
```

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

답 ②

함수 외부에 선언된 2번째 줄 int a, 3번째 줄 int b, 4번째 줄 int c 변수들은 전역변수이다.
func() 함수 내부에서 선언된 static int a와 int b는 지역변수이며, static int a는 지역변수이면서 정적변수이다.

▶ 전역변수(Global Variable)는 어디에서든 접근이 가능하고, main 함수 실행 전에 프로그램이 실행되자마자 메모리에 할당되고, 프로그램이 끝나는 순간 메모리에서 해제된다. 메모리의 **Data 영역에 적재**된다.
▶ 지역변수(Local Variable)는 변수가 선언된 해당 제어문이나 함수 등에서만 접근 가능하고, 함수가 실행되는 순간 메모리에 할당, 함수가 종료되는 순간 메모리에서 해제된다. 메모리의 **Stack 영역에 적재**된다.
▶ 정적변수(Static Variable) 변수가 선언된 해당 제어문이나 함수 등에서만 접근 가능한 것은 지역변수와 같지만, 단 한 번만 생성한 뒤 값이 계속 유지가 되며, main 함수 실행 전에 프로그램이 실행되자 마자 메모리에 할당되어, 프로그램이 끝나는 순간 메모리에서 해제된다. 전역변수와 마찬가지로 메모리의 **Data 영역에 적재**된다.

문제에서 처음에 프로그램이 실행될 경우, 전역변수 a, b, c와 func()에서 쓸 정적변수 a가 생성된다.
전역 a = 10, 전역 b = 20, 전역 c = 30, 정적 a = 100

main()함수가 실행되어 첫 번째 func()함수를 실행하면, 지역변수 b = 200가 생성된다.
그 이후 a++;에 의해 정적 a의 값이 1 증가해 101이 되고,
b++;에 의해 지역 b의 값이 1 증가해 201이 된다.
(**전역변수보다 함수 내부의 지역변수·정적변수가 우선**한다.)
그리고 c = a;에 의해 c에 101의 값이 들어간다.
(이 때 c는 지역변수가 없기 때문에 전역변수 c이다.)

그 이후 main()함수로 복귀해 두 번째 func()함수가 실행되고, 다시 b = 200이 생성된다.
이 때 정적변수는 기존의 값을 그대로 유지 101인 상태이다.
그리고 a++;에 의해 정적 a의 값이 1 증가해 102가 되고,
b++;에 의해 지역 b의 값이 1증가해 201이 된다.
그리고 c = a;에 의해 전역변수 c에 102의 값이 들어간다.

그 이후 main()함수로 복귀해 printf()문장이 실행되는데, 이 때의 출력값은 a, b, c 모두 전역변수의 값들이 출력된다.
전역 a와 전역 b는 바뀐 적이 없으므로 그대로 **a = 10**, **b = 20**이 출력되고, **c는 바뀐 대로 102의 값**이 출력된다.

문 20. 해싱(Hashing)에 대한 설명으로 옳지 않은 것은?

① 서로 다른 탐색키가 해시 함수를 통해 동일한 해시 주소로 사상될 수 있다.
② 충돌(Collision)이 발생하지 않는 해시 함수를 사용한다면 해싱의 탐색 시간 복잡도는 O(1)이다.
③ 선형 조사법(Linear Probing)은 연결리스트(Linked List)를 사용하여 오버플로우 문제를 해결한다.
④ 폴딩함수(Folding Function)는 탐색키를 여러 부분으로 나누어 이들을 더하거나 배타적 논리합을 하여 해시 주소를 얻는다.

답 ③

③ 연결리스트를 사용하여 오버플로우 문제를 해결하는 것은 체인법에 대한 설명이다.
▶ 체인이용법(Chaining): 충돌이 발생한 레코드들을 연결 리스트로 연결하는 방법으로, 해시 테이블 자체를 포인터 배열로 만들고 같은 버켓에 할당되는 레코드들을 체인으로 연결
▶ 선형조사법(Linear Probing): 충돌이 일어날 경우 데이터를 옆자리에 차례대로 저장
<오답 체크> ① 해시 함수가 서로 다른 두 개의 입력값에 대해 동일한 출력값을 출력하는 경우가 발생할 수 있는데, 이것을 **해시 충돌**(hash collision)이라고 한다.
② 충돌이 발생하지 않는다면, 찾고자 하는 자료를 해시 함수 연산 한 번에 바로 찾을 수 있다. 따라서 이 때의 시간 복잡도는 O(1)이 된다.
④ **폴딩함수**(Folding Function): 주로 탐색키가 해시 테이블의 크기보다 더 큰 정수일 경우에 사용하는 방법으로, 탐색키를 몇 개의 부분으로 나누어 이를 더하거나 비트별로 XOR 같은 부울 연산을 하는 방법