

2018-지방교행-컴퓨터일반-A형-해설-곽후근

1. 다음에서 설명되지 않은 기술은?

- 현실을 기반으로 가상 정보를 실시간으로 결합하여 보여주는 기술
- 인터넷을 기반으로 사물들을 연결하여 정보를 상호 소통하는 기술
- 많은 양의 정형 또는 비정형 데이터들로부터 가치를 추출하고 결과를 분석하는 기술
- 서로 다른 물리적인 위치에 존재하는 컴퓨터의 자원들을 가상화 기술로 통합해 제공하는 기술

- ① 빅데이터
- ② 블록체인
- ③ 사물인터넷
- ④ 증강현실

정답 체크 :

(2) 블록체인 : 관리 대상 데이터를 '블록'이라고 하는 소규모 데이터들이 P2P 방식을 기반으로 생성된 체인 형태의 연결고리 기반 분산 데이터 저장환경에 저장되어 누구라도 임의로 수정할 수 없고 누구나 변경의 결과를 열람할 수 있는 분산 컴퓨팅 기술 기반의 원장 관리 기술이다. 대표적으로 비트코인(가상화폐 또는 암호화폐)이 블록체인을 이용한다.

오답 체크 :

- (1) 빅데이터 : 많은 양의 정형 또는 비정형 데이터들로부터 가치를 추출하고 결과를 분석하는 기술이다.
- (3) 사물인터넷 : 인터넷을 기반으로 사물들을 연결하여 정보를 상호 소통하는 기술이다.
- (4) 증강현실 : 현실을 기반으로 가상 정보를 실시간으로 결합하여 보여주는 기술이다.

Tip! : 클라우드 컴퓨팅은 서로 다른 물리적인 위치에 존재하는 컴퓨터의 자원들을 가상화 기술로 통합해 제공하는 기술이다.

2. 다음 <조건>을 가장 만족하는 초당 표본추출(sampling) 횟수는?

- < 조건 >
- 디지털 신호로 변환하기 위한 아날로그 음성 신호의 최대 주파수는 4 KHz이다.
 - 표본추출된 디지털 신호를 아날로그 신호로 다시 변환할 경우 원래의 아날로그 음성 신호로 복원 되어야 한다.

- ① 1000
- ② 2000
- ③ 4000
- ④ 8000

정답 체크 :

(4)

해당 문제는 나이퀴스트 샘플링 이론(법칙)을 적용한다. 나이퀴스트 샘플링 이론이란 디지털 전송에서 부호 간 간섭을 없애는 조건으로 입력 신호의 최고(최대) 주파수 f_m 의 2배 이상의 주파수, 즉 $2f_m$ 이상의 주파수에서 표본화(샘플링)하면 원신호를 충실하게 재현할 수 있다는 정리이다. 그러므로 다음과 같은 계산이 가능하다.

초당 표본추출(샘플링, 표본화) = $2 \times \text{최대 주파수} = 2 \times 4\text{KHz} = 8\text{KHz} = 8000\text{Hz}$

3. 색상 모델(color model)에 대한 설명으로 옳은 것은?

- ① CMY 모델에서는 색을 혼합할수록 흰색에 가까워진다.
- ② RGB 모델은 감산 모델로 잉크젯 컬러 프린터에 이용된다.
- ③ HSV 모델은 사람의 직관적인 시각에 기초하여 색상, 채도, 명도로 구성된다.
- ④ RGB 모델로 표현된 색상은 CMY 모델로 표현된 색상으로 변환이 불가능하다.

정답 체크 :

(3) HSV : 인간의 직관적인 시각 모델과 흡사하고, 색상(Hue), 채도(Saturation), 명도(Value 또는 Brightness)로 구성된다.

오답 체크 :

- (1) CMY : 해당 설명은 RGB에 대한 설명이고, CMY는 색을 혼합할수록 검은색에 가까워진다.
- (2) RGB : 해당 설명은 CMY에 대한 설명이고, RGB는 가산 모델로 CRT 모니터에 사용된다.
- (4) 색상 모델(RGB, CMY, HSV)간의 변환은 가능하다.

4. 4-비트 2의 보수(2's complement)로 표현된 2진수 1110과 1010의 덧셈 결과를 10진수로 올바르게 표현한 것은?

- ① -8
- ② -4
- ③ 0
- ④ 8

정답 체크 :

(1)

$$1110 + 1010 = 1000$$

해당 덧셈 결과를 10진수로 바꾸는 방법은 2의 보수를 취하고(양수로 바꾸고) -를 붙이는 방법이 있다.

$$1000 \rightarrow 2의\ 보수(0111 + 1) \rightarrow 1000$$

여기에 음수를 붙이면 -8이 된다.

5. 인터넷에서 호스트네임(hostname)에 사상(mapping)되는 IP 주소를 찾기 위해 사용하는 것은?

- ① DNS(Domain Name System)
- ② OSPF(Open Shortest Path First)
- ③ ICMP(Internet Control Message Protocol)
- ④ SNMP(Simple Network Management Protocol)

정답 체크 :

(1) DNS : 도메인 혹은 호스트 이름을 IP 주소로 변환해 준다. Inverse DNS는 IP 주소를 도메인 혹은 호스트 이름으로 변환한다.

오답 체크 :

(2) OSPF : 라우팅 프로토콜로써 AS(autonomous system) 내부(Intra-AS)에서 경로배정을 위해 사용된다. 변경된 경로만 업데이트하기 때문에 traffic load를 감소시킨다.

(3) ICMP : 인터넷 제어 메시지 프로토콜은 RFC 792에서 정의한 인터넷 프로토콜 모음 중의 하나이다. ICMP 메시지들은 일반적으로 IP 동작에서 진단이나 제어로 사용되거나 오류에 대한 응답으로

만들어진다. 예를 들어, 핑(ping) 유틸리티는 ICMP "에코 요청(Echo request)"과 "에코 응답(Echo reply)" 메시지를 사용해 구현할 수 있다.

(4) SNMP : IP 네트워크상의 장치로부터 정보를 수집 및 관리하며, 또한 정보를 수정하여 장치의 동작을 변경하는 데에 사용되는 인터넷 표준 프로토콜이다. SNMP를 지원하는 대표적인 장치에는 라우터, 스위치, 서버, 워크스테이션, 프린터, 모뎀 랙 등이 포함된다. 예를 들어, 라우터의 상태를 보거나 아니면 어떤 설정을 필요로 할 때 SNMP를 이용할 수 있다.

6. 가상 메모리(virtual memory)에 대한 설명으로 옳은 것은?

- ① 세그멘테이션(segmentation) 기법을 적용할 수 없다.
- ② 프로세스를 주기억장치에 적재하지 않아도 프로세스의 수행을 가능하게 한다.
- ③ 주기억장치의 물리적 주소 공간과 사용자 관점의 논리적 주소 공간을 분리해 준다.
- ④ 페이징(paging) 과정에서 프레임이 필요 이상 할당 되면 스레싱(thrashing)이 발생한다.

정답 체크 :

(3) 사용자는 논리적 주소 공간을 바라보고 실제 사용할 때는 페이지 테이블을 통해 논리적 주소 공간과 물리적 주소 공간을 매핑한다.

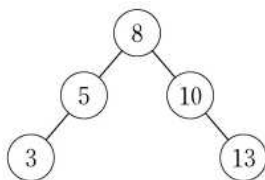
오답 체크 :

- (1) 가상 메모리의 대표적인 기법으로 페이징과 세그멘테이션 기법이 있다.
- (2) 프로세스를 주기억장치에 적재하지 않으면 프로세스를 수행할 수 없다.
- (4) 페이징 과정에서 프레임이 필요보다 적게 할당되면 스레싱(프로세스의 실행보다 페이지 교체에 더 많은 시간을 소요하는 비정상적인 현상)이 발생한다.

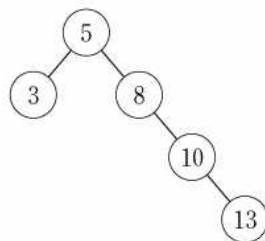
7. 다음과 같은 순서로 키 값들을 입력할 때 완성된 AVL 트리로 옳은 것은?

8 → 5 → 3 → 10 → 13

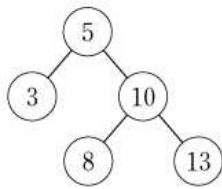
①



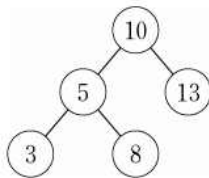
②



③



④



정답 체크 :

(3)

Adelson-Velskii와 Landis에 의해 1962년에 제안된 트리이다. 모든 노드의 왼쪽과 오른쪽 서브트리의 높이 차이가 1이하인 이진탐색트리이다. 트리가 비균형 상태로 되면 스스로 노드들을 재배치하여 균형 상태 유지한다. 균형 인수(balance factor)는 (왼쪽 서브 트리의 높이 - 오른쪽 서브 트리의 높이)를 의미하고, 모든 노드의 균형 인수가 ± 1 이하이면 AVL 트리이다.

AVL 트리의 균형이 깨지는 4가지 경우(삽입된 노드 N으로부터 가장 가까우면서 균형 인수가 ± 2 가 된 조상 노드가 A라면)는 다음과 같다.

LL 타입: N이 A의 왼쪽 서브 트리의 왼쪽 서브 트리에 삽입

LR 타입: N이 A의 왼쪽 서브 트리의 오른쪽 서브 트리에 삽입

RR 타입: N이 A의 오른쪽 서브 트리의 오른쪽 서브 트리에 삽입

RL 타입: N이 A의 오른쪽 서브 트리의 왼쪽 서브 트리에 삽입

각 타입별 재균형 방법은 다음과 같다.

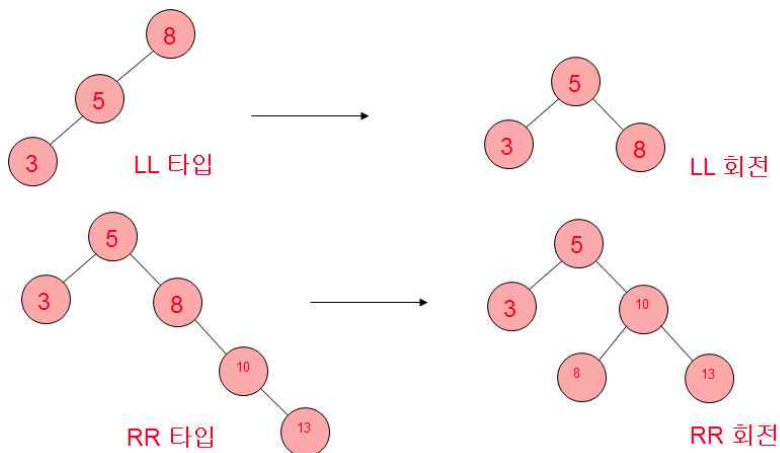
LL 회전: A부터 N까지의 경로상 노드의 오른쪽 회전

LR 회전: A부터 N까지의 경로상 노드의 왼쪽-오른쪽 회전

RR 회전: A부터 N까지의 경로상 노드의 왼쪽 회전

RL 회전: A부터 N까지의 경로상 노드의 오른쪽-왼쪽 회전

결론은 2진 탐색 트리처럼 삽입을 하다가 왼쪽 서브 트리와 오른쪽 서브 트리의 높이차가 2가되면 균형을 맞추기 위해서 회전을한다. 주어진 조건으로 문제를 풀면 다음과 같다. 8, 5, 3을 추가하게 되면 노드 8에서 균형 인수가 2가 되므로 LL 타입이 되고, LL 회전을 수행한다. 그리고 10, 13을 추가하게 되면 8에서 균형 인수가 -2가 되므로 RR 타입이 되고, RR 회전을 수행한다. 그러면 최종 AVL 트리가 완성된다.



8. 데이터베이스의 3단계-스키마 구조에 대한 설명으로 <보기>에서 옳은 것만을 모두 고른 것은?

< 보기 >

- ㄱ. 내부 스키마는 물리적 저장 장치의 관점에서 본 데이터베이스 구조이다.
- ㄴ. 외부 스키마는 각 사용자의 관점에서 본 데이터베이스 구조로서 여러 개가 존재할 수 있다.
- ㄷ. 개념 스키마는 모든 응용 시스템들이나 사용자 들이 필요로 하는 데이터를 통합한 조직 전체의 데이터베이스를 기술한 것이다.

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

정답 체크 :

(4)

(ㄱ) 내부 스키마 : 물리적인 저장 장치의 관점, 즉 전체 데이터베이스가 저장 장치에 실제로 저장되는 방법을 정의한 것이다. 레코드 구조, 필드 크기, 레코드 접근 경로 등 물리적 저장 구조를 정의한다.

(ㄴ) 외부 스키마 : 개별 사용자 관점, 즉 외부 단계에서 사용자에게 필요한 데이터베이스를 정의한 것이다. 각 사용자가 생각하는 데이터베이스의 모습, 즉 논리적 구조로 사용자마다 다르다. 서브 스키마(sub schema)라고도 한다.

(ㄷ) 개념 스키마 : 조직 전체의 관점, 즉 개념 단계에서 데이터베이스 전체의 논리적 구조를 정의한 것이다. 전체 데이터베이스에 어떤 데이터가 저장되는지, 데이터들 간에는 어떤 관계가 존재하고 어떤 제약조건이 존재하는지에 대한 정의뿐만 아니라, 데이터에 대한 보안 정책이나 접근 권한에 대한 정의도 포함한다.

9. 다음 C 프로그램의 출력값은?

```
#include <stdio.h>
void main(void)
{
    int i = 0, n = 167;
```

```

int b[8];
while (n > 0) {
    if (n%2 == 0) {
        b[i] = 0;
        n = n/2;
    }
    else {
        b[i] = 1;
        n = (n-1)/2;
    }
    i = i + 1;
}
for (i = 7; i >= 0; i--)
    printf("%d", b[i]);
}

```

- ① 10100001
- ② 10100111
- ③ 11100101
- ④ 11100111

정답 체크 :

(2)

i = 0, b[0] = 1, n = 83; // i가 0에서 n이 167이므로 2로 나눈 나머지가 1이 되어, b[0]에 1을 대입한다. 그리고 $n = (167-1) / 2 = 83$ 이 된다. i를 하나 증가시킨다.

i = 1, b[1] = 1, n = 41; // i가 1에서 n이 83이므로 2로 나눈 나머지가 1이 되어, b[1]에 1을 대입한다. 그리고 $n = (83-1)/2 = 41$ 이 된다. i를 하나 증가시킨다.

i = 2, b[2] = 1, n = 20; // i가 2에서 n이 41이므로 2로 나눈 나머지가 1이 되어, b[2]에 1을 대입한다. 그리고 $n = (41-1)/2 = 20$ 이 된다. i를 하나 증가시킨다.

i = 3, b[3] = 0, n = 10; // i가 3에서 n이 20이므로 2로 나눈 나머지가 0이 되어, b[3]에 0을 대입한다. 그리고 $n = 20/2 = 10$ 이 된다. i를 하나 증가시킨다.

i = 4, b[4] = 0, n = 5; // i가 4에서 n이 10이므로 2로 나눈 나머지가 0이 되어, b[4]에 0을 대입한다. 그리고 $n = 10/2 = 5$ 이 된다. i를 하나 증가시킨다.

i = 5, b[5] = 1, n = 2; // i가 5에서 n이 5이므로 2로 나눈 나머지가 1이 되어, b[5]에 1을 대입한다. 그리고 $n = (5-1)/2 = 2$ 이 된다. i를 하나 증가시킨다.

i = 6, b[6] = 0, n = 1; // i가 6에서 n이 2이므로 2로 나눈 나머지가 0이 되어, b[6]에 0을 대입한다. 그리고 $n = 2/2 = 1$ 이 된다. i를 하나 증가시킨다. 여기서 답이 나오므로 아래의 단계를 수행할 필요가 없다.

i = 7, b[7] = 1, n = 0; // i가 7에서 n이 1이므로 2로 나눈 나머지가 1이 되어, b[7]에 1을 대입한다. 그리고 $n = (1-1)/2 = 0$ 이 된다. n이 0이 되면 while 문을 탈출한다.

마지막 출력문을 보면 b[0]부터 출력하는 것이 아니라 b[7]부터 출력하는 것에 주의한다.

Tip! : 해당 코드는 10진수를 2진수로 변환하는 일반적인 코드로서 여러분들이 평소에 다양한 프로그래밍 코드를 접했다면 코드를 해석하지 않고도 문제를 빠르게 풀 수 있습니다.

10. 다음은 <질의>를 <SQL 문>으로 표현한 것이다. 빈칸 ㉠에 들어갈 내용으로 옳은 것은?

<질의>

사원 릴레이션에서 사원이 7명 이상인 부서에 대해서 부서명과 평균 급여를 구하시오. (단, 사원 릴레이션의 스키마는 (사원번호, 사원명, 부서명, 급여)이고, 기본키는 사원번호이다.)

<SQL 문>

SELECT 부서명, AVG(급여) FROM 사원 GROUP BY 부서명 ㉠ ;

- ① HAVING CHECK(*) >= 7
- ② HAVING COUNT(*) >= 7
- ③ WHERE CHECK(*) >= 7
- ④ WHERE COUNT(*) >= 7

정답 체크 :

(2) HAVING은 GROUP BY의 조건을 나타내기 위해 사용된다. 개수 계산(동일 부서의 사원수)을 위해 집계 함수 COUNT를 사용한다. *는 모든 컬럼(속성)을 대상으로 개수 계산을 수행하고, 컬럼 중에 비어 있는 값이 있다고 하더라도(예를 들어, 한 사원의 급여가 비어 있음) 개수 계산에 포함시키겠다는 의미이다.

오답 체크 :

- (1) CHECK는 CREATE TABLE 또는 ALTER TABLE에 제약 조건을 만들기 위해 사용된다.
- (3) WHERE는 SELECT의 조건을 나타내기 위해 사용된다. 문제의 조건은 SELECT의 조건이 아닌 GROUP BY의 조건을 나타낸다. CHECK는 CREATE TABLE 또는 ALTER TABLE에 제약 조건을 만들기 위해 사용된다.
- (4) WHERE는 SELECT의 조건을 나타내기 위해 사용된다. 문제의 조건은 SELECT의 조건이 아닌 GROUP BY의 조건을 나타낸다.

11. 부울(Boolean) 대수식 $A+B \cdot C$ 와 진리값이 다른 것은?

- ① $(A+B) \cdot (A+C)$
- ② $B \cdot C + A \cdot (A+B)$
- ③ $A+A \cdot C+B \cdot C$
- ④ $A+B \cdot (B+C)$

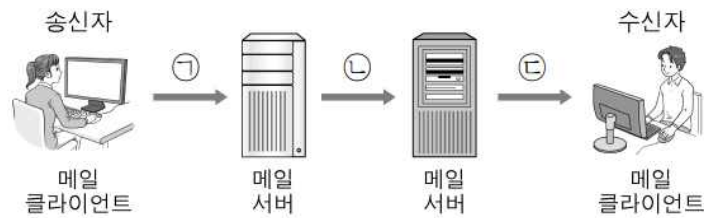
정답 체크 :

(4) $A+B(B+C) = A+B+BC = A+B(1+C) = A+B$

오답 체크 :

- (1) $(A+B)(A+C) = A+AC+AB+BC = A(1+AC+AB)+BC = A+BC$
- (2) $BC+A(A+B) = A+AB+BC = A(1+B)+BC = A+BC$
- (3) $A+AC+BC = A(1+C)+BC = A+BC$

12. 다음 그림은 전자우편의 전달 과정을 나타낸 것이다. ㉠~㉣에 사용되는 전자우편 프로토콜을 올바르게 짝지은 것은?



- | ㄱ | ㄴ | ㄷ |
|--------|------|------|
| ① SMTP | SMTP | IMAP |
| ② SMTP | IMAP | POP3 |
| ③ POP3 | SMTP | IMAP |
| ④ POP3 | IMAP | POP3 |

정답 체크 :

(1)

(ㄱ) : 메일 클라이언트가 메일 서버로 메일을 보낼 때 → SMTP 사용

(ㄴ) : 메일 서버가 메일 서버에게 메일을 보내거나 받을 때 → SMTP 사용

(ㄷ) : 메일 서버에서 메일 클라이언트가 메일을 내려 받을 때 → POP3(메일 사본을 메일 서버에 남기지 않음), IMAP(메일 사본은 메일 서버에 남김) 사용

13. 릴레이션 스키마 R(A, B, C, D, E, F)에서 함수적 종속이 다음과 같을 때, 제3 정규형을 만족 하도록 R을 분해한 것으로 옳은 것은? (단, R의 기본키는 (A, B)이다.)

{ (A, B) → C, B → D, A → E, E → F }

- ① R1(A, B, C, D)
R2(A, E)
R3(E, F)
- ② R1(A, B, C)
R2(B, D)
R3(A, E, F)
- ③ R1(A, B, C)
R2(B, D) R3(A, E) R4(E, F)
- ④ R1(A, C)
R2(B, C)
R3(B, D)
R4(A, E, F)

정답 체크 :

(3) 제3 정규형까지 모두 만족한다.

오답 체크 :

(1) R1(A, B, C, D) : 기본키인 (A, B)에 부분 함수 종속(B → D)되므로 제2 정규형을 만족하지 않습니다.

(2) R3(A, E, F) : 이행적 함수 종속(A → E, E → F)이 되어 제3 정규형을 만족하지 않습니다.

(4) R4(A, E, F) : 이행적 함수 종속(A → E, E → F)이 되어 제3 정규형을 만족하지 않습니다.

Tip! : 정규형이 높아질수록 분해의 수가 증가한다. 그러므로 (3) 또는 (4)가 답이 된다.

14. 모듈의 응집도에 대한 설명으로 <보기>에서 옳은 것만을 모두 고른 것은?

〈 보기 〉	
ㄱ. 모듈 내 한 구성 요소의 출력이 다른 구성 요소의 입력이 되는 경우는 순차적 응집도 (sequential cohesion)에 해당한다.	
ㄴ. 모듈 내 구성 요소들이 서로 다른 기능을 같은 시간대에 함께 실행하는 경우는 우연적 응집도 (coincidental cohesion)에 해당한다.	
ㄷ. 모듈이 여러 가지 기능을 수행하며 모듈 내 구성 요소들이 같은 입력 자료를 이용하거나 동일 출력 데이터를 만들어내는 경우는 통신적 응집도(communicational cohesion)에 해당한다.	

- ① ㄱ
 ② ㄴ
 ③ ㄱ, ㄷ
 ④ ㄴ, ㄷ

해설)

정답 체크 :

(3)

(ㄱ) 순차적 응집도 : A 요소의 출력을 B 요소의 입력으로 사용하므로 두 요소가 하나의 모듈을 구성한 경우이다. 두 요소가 아주 밀접하므로 하나의 모듈로 묶을 만한 충분한 이유가 된다.

(ㄷ) 통신적(교환적) 응집도 : 정보적 응집을 의미한다. 같은 입력을 사용하는 구성 요소들을 하나의 모듈로 구성한다. 구성 요소들이 동일한 출력을 만들어낼 때도 교환적 응집이라고 한다. 요소들 간의 순서는 중요하지 않다.

오답 체크 :

(ㄴ) 우연적 응집도 : 해당 설명은 시간적 응집도이고, 우연적 응집도는 구성 요소들이 말 그대로 우연히 모여 구성한다. 특별한 이유 없이, 크기가 커 몇 개의 모듈로 나누는 과정에서 우연히 같이 묶인 것이다.

Tip! : 응집도(cohesion)는 모듈 내부에 존재하는 구성 요소들 사이의 밀접한 정도이다. 하나의 모듈 안에서 구성 요소들 간에 뚝뚝 뭉쳐 있는 정도로 평가한다. 모듈 내 구성 요소 간의 응집도를 테이블로 정리하면 다음과 같다. 기능적이 좋은 품질이고, 우연적인 나쁜 품질이다.

기능적	함수적 응집을 나타낸다. 응집도가 가장 높은 경우이며 단일 기능의 요소로 하나의 모듈을 구성한다.
순차적	A 요소의 출력을 B 요소의 입력으로 사용하므로 두 요소가 하나의 모듈을 구성한 경우이다. 두 요소가 아주 밀접하므로 하나의 모듈로 묶을 만한 충분한 이유가 된다.
교환적	정보적 응집을 의미한다. 같은 입력을 사용하는 구성 요소들을 하나의 모듈로 구성한다. 구성 요소들이 동일한 출력을 만들어낼 때도 교환적 응집이라고 한다. 요소들 간의 순서는 중요하지 않다.
절차적	순서가 정해진 몇 개의 구성 요소를 하나의 모듈로 구성한다. 순차적 응집과 다른 점은 어떤 구성 요소의 출력이 다음 구성 요소의 입력으로 사용되지 않고, 순서에 따라 수행만 된다는 것이다.
시간적	모듈 내 구성 요소들의 기능도 다르고, 한 요소의 출력을 입력으로 사용하는 것도 아니고, 요소들 간에 순서도 정해져 있지 않다. 그러나 그 구성 요소들이 같은 시간대에 함께 실행된다는 이유로 하나의 모듈로 구성한다.

논리적	모듈 간 순서와 무관, 한 모듈의 출력을 다른 모듈의 입력으로 사용하는 것도 아니다. 그러나 요소들 간에 공통점이 있거나 관련된 임무가 존재하거나 기능이 비슷하다는 이유로 하나의 모듈로 구성한다.
우연적	구성 요소들이 말 그대로 우연히 모여 구성한다. 특별한 이유 없이, 크기가 커 몇 개의 모듈로 나누는 과정에서 우연히 같이 묶인 것이다.

15. 트리를 이용하여 표현할 수 있는 예로 가장 적절한 것은?

- ① 전국 고속도로 망
- ② 컴퓨터의 폴더 구조
- ③ 회전 초밥집의 회전대
- ④ 링(ring)형 네트워크 구조

정답 체크 :

(2) 트리를 이용하면 컴퓨터의 계층적 폴더들의 전체 크기를 계산할 수 있다.

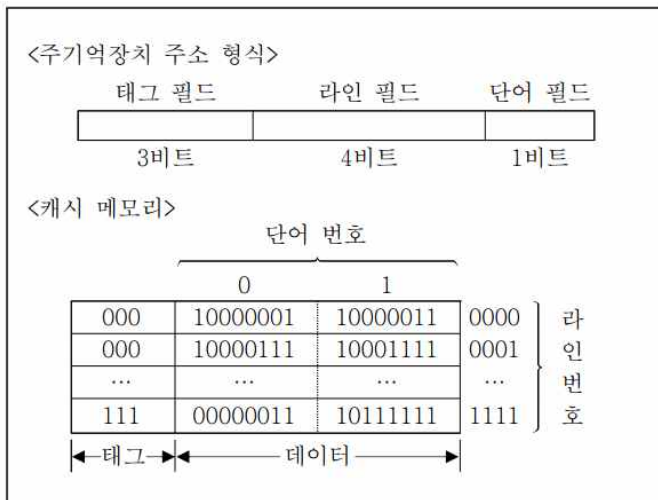
오답 체크 :

(1) 그래프를 이용하면 전국 고속도로 망을 표현할 수 있다.

(3) 원형 연결 리스트를 이용하면 회전 초밥집의 회전대를 표현할 수 있다. 임의의 위치에서 추가와 삭제가 가능하다.

(4) 원형 연결 리스트를 이용하면 링(ring)형 네트워크 구조를 표현할 수 있다. 임의의 위치에서 추가와 삭제가 가능하다.

16. 직접 사상(direct mapping) 방식을 사용하는 캐시 메모리와 주기억장치 주소 형식이 다음과 같을 때, 주기억장치 주소 00000011에 사상되는 데이터는? (단, 주기억장치는 바이트 단위로 주소가 지정된다.)



- ① 00000011
- ② 10000011
- ③ 10000111
- ④ 10001111

정답 체크 :

(4)

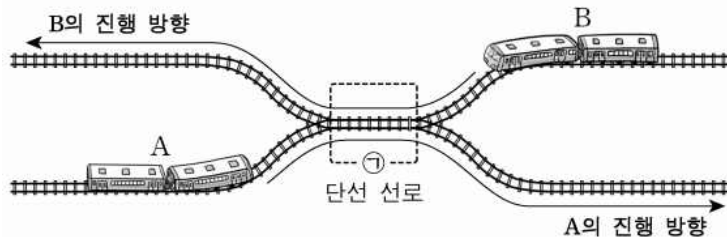
주어진 조건을 태그, 라인, 단어로 분리하면 다음과 같다.

라인 : 0001 // 캐시 메모리의 라인 번호에서 0001을 선택한다.

단어 : 1 // 라인 번호 0001에서 단어 번호 1을 선택한다.

태그 : 000 // 라인 번호 0001, 단어 번호 1의 태그 번호는 000이다.

17. 다음 그림은 컴퓨터에서의 교착상태를 기차길에서 발생할 수 있는 상황에 비유하여 나타낸 것이다. 이러한 교착상태 문제를 해결할 수 있는 방법으로 옳지 않은 것은?



- ① 기차 A, B가 모두 ㉠ 선로에 진입한 경우 은행원 알고리즘을 적용한다.
- ② ㉠ 선로를 우선적으로 사용할 수 있는 권한을 항상 A의 진행 방향으로 운행하는 기차에만 부여한다.
- ③ 모든 기차는 ㉠ 선로에 진입하기 전에 다른 방향에서 진입하는 기차가 없는 것을 기차 중앙 통제소를 통해 확인하고 진입한다.
- ④ ㉠ 선로 상에서 교착상태가 일어난 경우 기차 중앙 통제소는 기차 A, B 중 하나를 후진시켜 ㉠ 선로에 진입하기 이전으로 되돌린다.

정답 체크 :

(1) 두개의 기차가 모두 진입한 경우에는 탐지, 회복에 해당하는 방법을 사용해야 한다. 은행가 알고리즘은 회피에서 사용하는 방법이다.

오답 체크 :

- (2) 교착상태 해결 방법 중 예방에 해당한다.
- (3) 교착상태 해결 방법 중 회피에 해당한다.
- (4) 교착상태 해결 방법 중 탐지, 회복에 해당한다.

18. CIDR(Classless Inter-domain Routing) 주소지정(addressing)에서 포워딩(forwarding) 테이블이 다음과 같을 때, 목적지 IP 주소 123.4.8.16에 해당하는 출력 인터페이스는?

네트워크 주소 / 마스크	출력 인터페이스
123.4.8.128 / 25	m1
123.4.8.0 / 24	m2
123.4.0.0 / 16	m3
0.0.0.0 / 0	m4

- ① m1
- ② m2
- ③ m3
- ④ m4

정답 체크 :

(2)

네트워크 주소와 마스크를 이용한 사용 가능 주소는 다음과 같다.

123.4.8.128/25 : 123.4.8.128 ~ 123.4.8.255 : m1

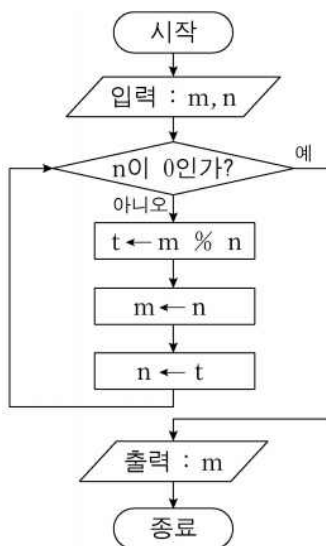
123.4.8.0/24 : 123.4.8.0 ~ 123.4.8.255 : m2

123.4.0.0/16 : 123.4.0.0 ~ 123.4.255.255 : m3

0.0.0.0/0 : 0.0.0.0 ~ 255.255.255.255 : m4

포워딩 테이블은 목적지 IP 주소에 해당하는 네트워크 주소를 찾는 것인데 일치하는 것이 여러 개 있다면 범위가 작은 것을 우선으로 한다. m2, m3, m4 모두가 해당되지만 이중 범위가 가장 작은 것은 m2이다.

19. 다음 순서도에서 m에 104, n에 39를 입력할 때 출력값은? (단, % 는 나머지 연산자이다.)



- ① 0
- ② 2
- ③ 13
- ④ 26

정답 체크 :

(3)

m = 104, n = 39 // 문제의 주어진 조건이다. n이 0이 아니므로 아래로 진행한다.

t = 104 % 39 = 26, m = 39, n = 26 // t, m, n이 수정된다. n이 0이 아니므로 아래로 진행한다.

t = 39 % 26 = 13, m = 26, n = 13 // t, m, n이 수정된다. n이 0이 아니므로 아래로 진행한다.

t = 26 % 13 = 0, m = 13, n = 0 // t, m, n이 수정된다. n이 0이 종료하고 m을 출력한다.

20. 다음 C 프로그램의 출력값은?

```
#include <stdio.h>
#define SIZE 7
```

```

void main(void)
{
    int i, tmp;
    int num[SIZE] = {58,28,81,98,16,64,70};
    for (i = 0; i < SIZE / 2; i++) {
        if (num[i] < num[SIZE - 1 - i]) {
            tmp = num[i];
            num[i] = num[SIZE - 1 - i];
            num[SIZE - 1 - i] = tmp;
        }
    }
    for (i = 0; i < SIZE; i++)
        printf("%d ", num[i]);
}

```

- ① 58 28 81 98 16 64 70
- ② 58 28 98 81 64 70 16
- ③ 70 64 16 98 81 28 58
- ④ 70 64 81 98 16 28 58

정답 체크 :

(4)

i = 0; num[0] (58)보다 num[6] (70)이 크므로 서로 교환한다.

i = 1; num[1] (28)보다 num[5] (64)이 크므로 서로 교환한다.

i = 2; num[2] (81)보다 num[4] (16)이 크지 않으므로 서로 교환하지 않는다.

for 문 조건은 3(= SIZE / 2 = 7 / 2, 소수점 이하 버림) 전인 2까지 반복하는 것이므로 더 이상 진행하지 않는다.

Tip! : 문제의 조건만 이해하면 더 이상 프로그래밍 문제가 아니다 즉, 기계적으로 계산이 가능한 문제이다.