2018년 국회직 9급 컴퓨터일반 풀이

by 호이호이꿀떡

정답 체크

01	02	03	04	05	06	07	08	09	10
①	③	②	⑤	⑤	①	④	③	②	①
11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
①	④	②	③	③	②	⑤	①	⑤	④

1. 데이터베이스에서 트랜잭션에 대한 설명 중 다음의 특성이 의미하는 것으로 옳은 것은?

> 각 트랜잭션은 다른 트랜잭션과 독립적으로 수행되는 것처럼 보여야 하며, 다른 트랜잭션에 영향을 미치지 않는다.

① 고립성(Isolation)
② 일관성(Consistency)
③ 원자성(Atomicity)
④ 지속성(Durability)
⑤ 투명성(Transparency)

답 ①

◈ **트랜잭션의 특성**(ACID)
▷ **원자성**(Automicity)
트랜잭션에 포함된 오퍼레이션(작업)들은 모두 수행되거나, 아니면 전혀 수행되지 않아야 한다.
▷ **일관성**(Consistency)
트랜잭션이 성공적으로 완료되면, 일관성있는 상태로 있어야 한다.
▷ **고립성, 독립성**(Isolation)
각 트랜잭션은 독립적으로 수행되고, 실행 중 다른 트랜잭션이 끼어들지 않아야 한다.
▷ **지속성**(Durability) : 성공적으로 수행된 트랜잭션의 결과는 계속해서 유지되어야 한다.

2. 프로세스의 상태 변이에 대한 설명으로 옳지 않은 것은?

① 시간 할당량(time slice)을 사용하는 일반적 우선순위 기반 스케줄링에서 실행(running) 상태 프로세스의 시간 할당량이 모두 소진되었을 때, 우선순위가 높은 다른 준비 상태의 프로세스가 있다면 실행 중이던 프로세스는 커널(kernel)에 의해 스케줄링되기를 기다리는 준비(ready) 상태로 전이된다.
② 실행 상태의 프로세스가 동기식 입출력 요청을 하면, 일반적으로 해당 프로세스는 입출력이 완료될 때까지 CPU를 반납하고 대기(blocked 또는 waiting) 상태로 전이된다.
③ 대기 상태의 프로세스가 요청하였던 입출력이 완료되면, 해당 프로세스는 CPU 연산이 가능해지므로 바로 실행 상태로 전이된다.
④ 다중 처리기 시스템(multi-processor system)에서는 실행 상태의 프로세스가 여러 개 있을 수 있다.
⑤ 대기 상태의 프로세스들은 CPU 할당을 위한 스케줄링에서 제외된다

답 ③

③ 대기 상태의 프로세스가 입출력이 완료되면 바로 실행 상태로 전이되는 것이 아니라, 준비 상태로 전이되어 자신의 순서가 오길 기다린다.

<오답 체크> ⑤ 대기 상태의 프로세스에는 우선순위가 부여되지 않으므로, 스케줄링에서 제외된다.

3. OSI(Open System Interconnection) 7계층에 대한 설명으로 옳지 않은 것은?

① 7계층에서 물리적 전송 매체와 직접적으로 연관이 있는 계층은 물리 계층이다.
② 수신 측에서 패킷을 수신하게 되면, 상위 계층에서 하위 계층 순으로 처리된다.
③ 전송 계층(transport layer)의 대표적인 프로토콜로는 TCP, UDP 등이 있다.
④ 네트워크 계층에서는 라우팅 및 패킷 전송의 기능을 수행한다.
⑤ 응용 계층에서는 전자 사서함, 파일 전송 등의 서비스를 제공한다.

답 ②

② 송신 측에서 패킷을 송신할 때 상위 계층(응용 계층)에서 하위 계층(물리 계층) 순으로 처리되고, 수신 측에서 패킷을 수신할 때는 하위 계층에서 상위 계층 순으로 처리된다.

<오답 체크>

③ 3계층 네트워크 계층 : IP, ARP, RARP, ICMP, IGMP 등
4계층 전송 계층 : TCP, UDP 등
7계층 응용 계층 : FTP, TELNET, HTTP, SMTP, POP, IMAP 등

4. 다음 정렬 알고리즘의 수행시간을 Big-O 표기법으로 나타냈을 때 최악의 경우의 수행시간이 같은 것들로만 나열된 것은?

선택 정렬(selection sort), 합병 정렬(merge sort),
삽입 정렬(insertion sort), 퀵 정렬(quick sort),
힙 정렬(heap sort)

① 합병 정렬, 퀵 정렬, 힙 정렬
② 힙 정렬, 선택 정렬, 퀵 정렬
③ 합병 정렬, 선택 정렬, 삽입 정렬
④ 합병 정렬, 힙 정렬, 삽입 정렬
⑤ 선택 정렬, 삽입 정렬, 퀵 정렬

답 ⑤

⑤ 퀵 정렬의 평균 시간복잡도는 $O(n\log_2 n)$으로 합병 정렬, 힙 정렬과 같지만, 최악일 때의 시간복잡도는 $O(n^2)$으로 선택 정렬, 삽입 정렬과 같다.

◈ 정렬 알고리즘 시간복잡도(Time complexity)

알고리즘	최선	평균	최악
삽입 정렬	$O(n)$	$O(n^2)$	$O(n^2)$
선택 정렬	$O(n^2)$	$O(n^2)$	$O(n^2)$
버블 정렬	$O(n^2)$	$O(n^2)$	$O(n^2)$
셸 정렬	$O(n)$	$O(n^{1.5})$	$O(n^2)$
퀵 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n^2)$
힙 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
합병 정렬	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$
기수 정렬	$O(dn)$	$O(dn)$	$O(dn)$

5. 관계형 데이터베이스(relational database)에 대한 설명으로 옳은 것을 <보기>에서 모두 고르면?

〈 보 기 〉

ㄱ. 스키마 변환 시 정보의 무손실, 자료 중복의 감소, 관련된 구조 간의 통합의 원칙을 준수하여야 한다.

ㄴ. 관계대수(relational algebra)의 연산에서 피연산자는 모두 릴레이션이지만 연산결과는 릴레이션이 아니다.

ㄷ. 릴레이션에 연산을 수행 시 삽입이상(insertion anomaly), 삭제이상(deletion anomaly), 갱신이상(update anomaly)이 발생할 수 있다.

ㄹ. 튜플을 구성하는 속성 사이에 존재하는 종속관계를 고려하지않고 하나의 릴레이션으로 표현하여 이상(anomaly)을 해결할 수 있다.

ㅁ. 릴레이션이 여러 속성을 표현할 때 이를 작게 분해(decomposition)하는 과정을 정규화(normalization)라고 한다.

ㅂ. 릴레이션들은 관계대수(relational algebra)로 조작이 가능하다.

① ㄱ, ㄴ, ㄷ
② ㄴ, ㄷ, ㄹ
③ ㄴ, ㄹ, ㅁ
④ ㄷ, ㄹ, ㅂ
⑤ ㄷ, ㅁ, ㅂ

답 ⑤

<오답 체크> ㄱ. 스키마 변환의 원칙 - 정보의 무손실, 데이터 중복의 최소화, 분리의 원칙

ㄴ. 관계대수는 릴레이션을 처리하기 위해 연산자와 연산규칙을 제공하는 언어로 피연산자가 릴레이션이고 결과도 릴레이션이다.

ㄹ. 속성들 간의 종속성을 분석해서 하나의 릴레이션에는 하나의 종속성이 표현되도록 분해해야 한다.

6. 후위표기법(postfix notation)으로 된 다음 식의 전위표기법(prefix notation)으로 옳은 것은?

ABC+D/-AE+BF*/+

① +-A/+BCD/+AE*BF
② -+A/BC+D/+AE*BF
③ +-A/+BCD/+*AEBF
④ +A-/+BCD/+AE*BF
⑤ -+/A+BCD/+*AEBF

답 ①

1) ABC+D/- AE+BF*/+ => A+BCD/- AE+BF*/+
2) A+BCD/- AE+BF*/+ => A/+BCD- AE+BF*/+
3) A/+BCD- AE+BF*/+ => -A/+BCD AE+BF*/+
4) -A/+BCD AE+BF*/+ => -A/+BCD +AEBF*/+
5) -A/+BCD +AEBF*/+ => -A/+BCD +AE*BF/+
6) -A/+BCD +AE*BF/+ => -A/+BCD /+AE*BF+
7) -A/+BCD /+AE*BF+ => +-A/+BCD /+AE*BF

7. 다음 카르노 맵에 해당하는 논리식 F로 옳은 것은?

A / BC	0	1
00	0	1
01	0	0
11	1	0
10	1	0

① F = AB + AB′C
② F = A′B′ + A′B′C′
③ F = AB′ + A′BC
④ F = A′B + AB′C′
⑤ F = AB + BC

답 ④

④ 왼쪽 아래 두 개(**파란색**)를 묶으면 A`B 가 되며,
오른쪽 위 하나(**빨간색**)는 AB`C` 이다.
따라서 논리식 F는 A`B + AB`C` 가 된다.

A / BC	0	1
00	0	1
01	0	0
11	1	0
10	1	0

8. 가상 메모리 시스템에서 메모리 부족 시의 페이지 교체 기법(page replacement algorithm)들에 대한 설명으로 옳지 않은 것은?

① LRU(Least Recently Used) 기법은 메모리에 적재된 페이지 중 가장 오랫동안 참조되지 않았던 페이지를 교체하는 기법이다.

② LRU 기법은 실제 그 구현 오버헤드가 커서, 일반적으로 오버헤드를 줄인 여러 유형의 LRU 근사 알고리즘(LRU approximation algorithm)들이 사용되는 것이 보통이다.

③ LRU 기법은 물리적 페이지의 개수를 확장했음에도 페이지 폴트가 늘어나는 경우가 발생할 수도 있는데, 이를 Belady's anomaly라 한다.

④ 이론적으로는 최적의 페이지 교체 기법은 메모리에 적재된 페이지들 중에서 앞으로 가장 오랫동안 참조되지 않을 페이지를 교체하는 것이다.

⑤ FIFO(First In First Out) 기법은 메모리에 적재된 페이지들 중 가장 먼저 메모리에 적재된 페이지를 교체하는 방법이다.

답 ③

③ 벨라디의 모순(Belady`s anomaly)이 발생하는 알고리즘은 FIFO 기법이다.

◈ 페이지 교체 알고리즘

▶ **OPT**(OPTimal replacement, 최적 교체)
- 앞으로 가장 오랫동안 사용하지 않을 페이지를 교체하는 기법
- 각 페이지의 호출 순서와 참조 상황을 미리 예측해야 하므로 실현 가능성이 희박

▶ **FIFO**(Firstt In First Out)
- 가장 먼저 들어와서 가장 오래 있었던 페이지를 교체하는 기법
- 벨레이디의 모순현상(Belady's Anomaly)이 발생
  [일반적으로 페이지 프레임 수가 많으면 페이지 부재의 발생 횟수가 줄어들지만, 페이지 프레임 수를 증가시켰는데도 불구하고 페이지 부재 횟수가 더 많이 일어나는 현상]

▶ **LRU**(Least Recently Used)
- 최근에 가장 오랫동안 사용하지 않은 페이지를 교체하는 기법
- 계수기나 스택과 같은 별도의 하드웨어가 필요하며, 시간적인 오버헤드가 발생

▶ **LFU**(Least Frequently Used)
- 사용 빈도가 가장 적은 페이지를 교체하는 기법
- 프로그램 실행 초기에 많이 사용된 페이지가 그 후로 계속 페이지 프레임을 점유

▶ **NUR**(Not Used Recently)
- 최근에 사용하지 않은 페이지를 교체하는 기법
- 최근 사용 여부를 확인하기 위해서 각 페이지마다 참조 비트(Reference Bit)와 변형 비트(Modified Bit, Dirty Bit)를 사용
- 참조하지 않은 페이지를 우선 교체

▶ **MRU**(Most Recently Used)
- 가장 최근에 사용했던 페이지를 교체하는 기법(LRU의 반대)
- 특정한 반복 순환 구조 효율적

▶ **MFU**(Most Frequently Used)
- 사용 빈도가 가장 많은 페이지를 교체하는 기법
- 적게 사용된 페이지가 방금 들어온 것이고 아직 덜 사용되었으므로, 앞으로 사용될 확률이 높다는 계산에 근거

▶ **SCR**(Second Chance Replacement, 2차 기회 교체)
FIFO 기법의 단점을 보완하는 기법으로, 오랫동안 주기억장치에 있었지만 자주 사용되는 페이지의 교체를 방지하기 위한 기법이다.
각 페이지마다 참조 비트를 두어, 교체 대상이 되기 전에 참조 비트를 검사하여 1일 경우, 교체는 하지 않고 참조 비트만 0으로 바꾸어 한번의 기회를 더 부여한다. 그 다음에 참조되지 않아 참조 비트가 0으로 남아있는 상태에서 또 교체해야 할 상황이 오면 교체한다.

9. 요즘 집안에서 스마트폰 및 컴퓨터의 인터넷 접속을 위하여 공유기의 사용이 일반화되어 있다. 공유기에 접속하는 컴퓨터 디바이스에는 사설 IP가 주로 할당되는데 이때 IP주소의 자동 할당을 위해 사용되는 프로토콜로 옳은 것은?

① ARP
② DHCP
③ TCP
④ UDP
⑤ ICMP

답 ②

② **DHCP**(Dynamic Host Configuration Protocol, 동적 호스트 구성 프로토콜)
조직 내의 네트워크 상에서 IP 주소를 중앙에서 관리하고 할당해주는 프로토콜이다. 네트워크에 연결된 모든 기기가 각각 고유의 IP주소를 가지는 게 아니라, 네트워크 접속이 필요할 때만 IP 주소를 할당 받아 사용하기 때문에 IP 주소의 낭비를 줄일 수 있다.

<오답 체크>

① **ARP**(Address Resolution Protocol, 주소 결정 프로토콜)
IP 주소를 대응하는 MAC 주소값로 변환해주는 프로토콜

③ **TCP**: OSI 4계층 전송계층에서 작동하는 신뢰성, 연결 지향형의 프로토콜. 전이중 전송방식/양방향성(full-duplex)의 특성을 가지고 있으며, 흐름제어, 혼잡제어 등의 기능을 수행한다.

④ **UDP**: 비연결형으로 신뢰성을 보장하지 않는대 대신, 빠른 전송을 위해 사용하는 프로토콜

⑤ **ICMP**(Internet Control Message Protocol, 인터넷 제어 메시지 프로토콜)
호스트와 게스트 간의 통신 중에 발생하는 에러 및 제어 정보를 제어하는 프로토콜

10. 다음의 C 프로그램 실행 결과로 출력되는 a, b, c 값으로 옳은 것은?

```
#include <stdio.h>
int foo(int a, int *b)
{
    int c;
    *b = a + 1;
    c = a - 1;
    return c;
}
void main()
{
    int a = 5;
    int b = 3;
    int c = 0;
    b = foo(a, &c);
    c = foo(b, &a);
    printf("a=%d b=%d c=%d\n", a, b, c);
}
```

① 5, 4, 3
② 4, 3, 2
③ 6, 3, 4
④ 4, 3, 5
⑤ 5, 3, 0

답 ①

(foo 함수 안의 a, b, c와 main 함수 안의 a, b, c를 헷갈리지 않도록 주의한다.)
foo 함수를 뜯어보면 두 개의 변수를 입력받아, 앞의 변수에 1을 더한 수를 뒤의 변수에 집어넣고, 앞의 값에서 1을 뺀 수를 반환한다.
먼저 b = foo(a, &c);를 실행하면, a에 1을 더한 값(**6**)을 c에 입력하고, a에서 1을 뺀 값(**4**)을 b로 반환 받는다.
결과 : a = 5, b = 4, c = 6

그 다음 c = foo(b, &a);를 실행하면, b에 1을 더한 값(**5**)를 a에 입력하고, b에서 1을 뺀 값(**3**)을 c로 반환 받는다.
결과 : a = 5, b = 4, c = 3

11. 데이터베이스에서 사용하는 뷰(view)에 대한 설명으로 옳은 것을 <보기>에서 모두 고르면?

〈 보 기 〉
ㄱ. 뷰의 정의는 변경할 수 없다.
ㄴ. 뷰는 삽입, 갱신, 삭제 연산에 제약이 있다.
ㄷ. 뷰 위에 다른 뷰를 정의할 수 없다.
ㄹ. 뷰가 정의된 테이블이 삭제되더라도 뷰는 남는다.
ㅁ. 뷰는 물리적으로 구현되는 테이블이다.

① ㄱ, ㄴ
② ㄱ, ㅁ
③ ㄴ, ㄷ
④ ㄷ, ㄹ
⑤ ㄹ, ㅁ

답 ①

ㄱ. 뷰의 정의는 ALTER문을 이용해 변경할 수 없으며, 수정하고 싶으면 삭제 후 다시 정의해야 한다.
ㄴ. 뷰에 대한 검색은 기본 테이블과 거의 동일하지만, 삽입, 갱신, 삭제 연산에는 제약이 있다.
<오답 체크> ㄷ. 뷰 위에 다른 뷰를 정의할 수 있다.
ㄹ. 뷰가 정의된 테이블이 변경되면 뷰도 변경되며, 테이블이 삭제되면 뷰도 삭제된다.
ㅁ. 뷰는 논리적으로 구현되는 테이블이다.

12. 프로세스(process)나 스레드(thread)들이 공유 자원에 하나 이상의 수정(write 또는 modify) 연산을 포함하는 동시 접근을 할 때 그 접근 부분들을 임계 구역(critical section)이라 한다. 이를 보호하기 위한 병행 프로세스 동기화 기법으로 옳은 것은?

① 인터럽트(interrupt)
② 선점 스케줄링(preemptive scheduling)
③ 문맥교환(context switching)
④ 상호배제(mutual exclusion)
⑤ 교착상태(deadlock)

답 ④

④ **상호배제**(mutual exclusion)
특정 프로세스가 공유 자원을 사용 중일 때 다른 프로세스가 이 자원에 접근하지 못하도록 막는 것을 의미한다.
<오답 체크> ① **인터럽트**(interrupt)
마이크로프로세서가 프로그램을 실행하고 있을 때, 입출력 하드웨어 등의 장치나 또는 예외상황이 발생하여 처리가 필요할 경우에 마이크로프로세서에게 알려 처리하도록 요청하는 것을 말한다.
② **선점 스케줄링**(preemptive scheduling)
한 프로세스가 실행 중 더 높은 우선순위의 프로세스가 들어왔을 때, 현재 실행중인 프로세스로부터 강제로 CPU를 빼앗아 새로 들어온 프로세스에게 할당할 수 있는 스케줄링 기법을 말한다.
③ **문맥교환**(context switching)
하나의 프로세스가 CPU를 사용 중인 상태에서 다른 프로세스가 CPU를 사용하도록 하기 위해, 이전의 프로세스의 상태(문맥)를 보관하고 새로운 프로세스의 상태를 적재하는 작업을 말한다.
⑤ **교착상태**(deadlock)
두 개 이상의 작업이 서로 상대방의 작업이 끝나기만을 기다리면서 결과적으로 아무것도 완료되지 못하는 상태를 말한다.

13. 배열 int array[10][200]를 행우선순서(row major order)로 저장하는 경우의 원소 array[7][12]의 시작주소는 몇 번지인가?
(단, 배열 array의 시작주소는 10840h로, int의 크기는 4바이트로 가정한다. 배열 첨자는 0부터 시작하며 숫자에 붙은 h는 16진수 표기를 의미한다.)

① 10804h
② 11E50h
③ 16488h
④ 108BFh
⑤ 10A3Ch

답 ②

행우선순서로 저장할 경우, array[0] 행의 값들을 모두 저장한 뒤 a[1] 행의 값들을 저장하고, array[1] 행의 값들을 모두 저장한 뒤 a[2] 행의 값들을 저장한다.
먼저 array[0][0]~array[0][199]까지 200개를 저장한 뒤,
array[1][0]부터 array[1][199]까지 다음 200개를 저장하고,
또 array[2][0]부터 array[2][199]까지 다음 200개를 저장하는 방식이다.
이렇게 array[6][199]까지 1400개를 저장한다.
그 다음 array[7][0]부터 array[7][12]까지 13개이며, 따라서 array[7][12]는 1413번째 자리에 저장되는 원소이다. 그리고 이것은 array의 시작지점에서 1412칸 떨어진 위치이며, 한 칸이 4바이트이므로, 시작주소인 10840h에서 5648바이트만큼 떨어진 자리이다.
이것을 식으로 표현하면,

200 * 7 + 12 = 1412
1412 * 4 = 5648

문제에서 주소는 16진수로 표현되므로, 5648를 16진수로 변환하면 1610h가 된다.

16) 5648
16) 353 --- 0
16) 22 --- 1
16) 1 --- 6
0 --- 1

따라서 최종 주소값은 10840h + 1610h = 11E50h

14. 다음은 마트에서 판매 기록을 저장한 트랜잭션(transaction) 데이터이다. 규칙 "기저귀" -> "맥주"의 신뢰도(confidence) 값으로 옳은 것은?

식별자	품목
1	맥주, 땅콩, 기저귀
2	맥주, 커피, 기저귀
3	맥주, 기저귀, 계란
4	땅콩, 계란, 우유
5	땅콩, 커피, 기저귀, 우유

① 0.33
② 0.5
③ 0.75
④ 0.8
⑤ 1.00

답 ③

데이터 연관성 분석에서 A -> B 신뢰도는 A가 포함된 거래 중에서 B가 포함된 거래의 비율을 의미한다.
기저귀가 포함된 거래 1, 2, 3, 5 중 맥주가 포함된 거래 1, 2, 3 이다. 따라서 신뢰도는 3/4 = 0.75 = 75% 가 된다.

15. 사용자가 WWW(World Wide Web)을 사용하여 임의의 사이트에 접속하는 과정의 설명으로 옳지 않은 것은?

① 사용자의 웹브라우저에서 DNS 서버로 DNS 질의 메시지를 보낸다.
② DNS 간의 질의는 재귀형(recursive) 또는 반복형(iterative)으로 연계된다.
③ 사용자의 웹브라우저는 DNS 서버로부터 사이트의 웹 서버 MAC 주소를 획득한다.
④ HTTP 요청 메시지를 보내기 전에 TCP 3-방향 핸드셰이크를 수행한다.
⑤ 웹 서버가 보낸 HTTP 응답 메시지의 바디로부터 HTML을 추출하여 웹 페이지를 보여준다.

답 ③

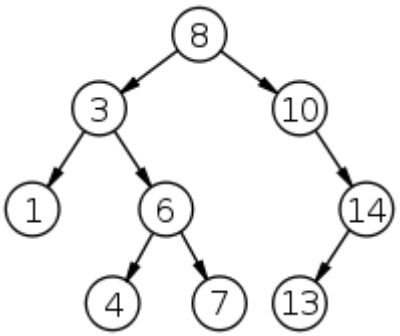
③ 사용자의 웹브라우저는 DNS 서버로부터 사이트의 IP 주소를 획득한다.

16. 1과 1000 사이의 정수로 구성된 이진탐색트리(binary search tree)에서 숫자 573을 탐색하는 경우 다음 중 비교경로로 옳지 않은 것은?

① 2, 173, 241, 856, 301, 489, 710, 516, 573
② 7, 816, 68, 714, 121, 561, 278, 395, 573
③ 981, 825, 693, 38, 137, 608, 224, 461, 573
④ 926, 139, 884, 278, 734, 319, 662, 481, 573
⑤ 14, 970, 831, 765, 111, 249, 318, 473, 573

답 ②

이진탐색트리는 아래와 같이 한 노드를 기준으로 왼쪽 서브트리에는 작은 값들이, 오른쪽 서브트리에는 큰 값들이 모여있는 구조이다.
그러므로 한 노드에서 왼쪽(오른쪽)으로 갔다는 것은 그 노드를 기준으로 찾는 값이 작다는(크다는) 것을 의미하며, 다시 그 값보다 큰 값(작은 값)을 찾는 일은 없어야 한다.



②번을 보면, 6번째에 561과 비교하고 있다. 문제에서 찾는 값은 573이기 때문에, 561에서 오른쪽 서브트리로 이동해야 하며 561보다 작은 값과는 비교할 일이 없어야 한다. 그런데 561 다음에 278과 비교하고 있는데, 이는 561의 왼쪽 서브트리로 이동하였다는 말이 된다. 이것이 첫 번째 오류이다.
두 번째 오류는 561보다 작은 값들을 비교하러 왼쪽 서브트리로 내려갔으므로, 561의 오른쪽 서브트리에 있는 큰 수인 573을 찾을 수 없어야 한다. 하지만 문제에서는 마지막에 573을 찾아갔고, 이것이 두 번째 오류이다.

17. 유니버설 게이트 집합(universal gate set)은 그 구성 원소만으로 어떤 형태의 디지털 시스템도 구현할 수 있는 기능적으로 완전한 게이트들의 집합이다. 다음 중 유니버설 게이트 집합으로 옳지 않은 것은?

① { NAND }
② { OR, DECODER }
③ { MULTIPLEXER }
④ { OR, NOT }
⑤ { XOR }

답 ⑤

⑤ XOR 게이트는 유니버셜 게이트가 될 수 없다.

<오답 체크> ③ 일반적으로 이론서에는 NAND 게이트나 NOR 게이트만 유니버셜 게이트라고 나와있다. 하지만 멀티플렉서(MULTIPLEXER) 역시 입력선 A, B에 0이나 1의 입력선을 추가해 어떻게 순서를 배치하느냐에 따라 모든 게이트를 표현할 수 있다.

예를 들어 입력선 x, y, z에 대해 멀티플렉서의 식을 (xz` + yz)로 설정할 경우, x에 0을 입력, y에 B를 입력, z에 A를 입력하면 출력값 F = 0·A` + B·A = A·B가 되어 AND 게이트가 된다.

반면 x에 B를 입력, y에 1을 입력, z에 A를 입력하면 출력값 F = B·A` + 1·A = B·A` + A = (B+A)·(A`+A) = A + B가 되어 OR 게이트가 된다.

②④ OR 게이트 하나로는 유니버셜 게이트가 될 수 없지만, OR 게이트와 NOT 게이트를 활용하면 NOR 게이트를 만들 수 있기 때문에 OR와 NOT의 집합은 유니버셜 게이트 집합이 된다. 디코더(DECODER)로 NOT 역할만 하도록 만들면 OR와 합쳐 NOR 게이트와 같도록 만들 수 있다.

18. 다음 C 프로그램의 실행 결과로 옳은 것은?

```
#include 〈stdio.h〉

int recursion(int n)
{
    if (n 〈 5) return 1;
    else if (n % 5 == 1) return n + recursion(n - 1);
    else recursion(n - 1);
}

int main()
{
    int n = recursion(16);
    printf("%d", n);

    return 0;
}
```

① 34
② 33
③ 31
④ 29
⑤ 28

답 ①

recursion(16)을 실행하면 5보다 작지 않고 5로 나눈 나머지가 1이기 때문에 recursion 함수 내에서 두 번째 줄이 실행된다.
따라서 16 + recursion(15)를 실행한다.
여기서 다시 recursion(15)는 recursion 함수 내에서 세 번째 줄이 실행되어 recursion(14)를 실행한다.
이렇게 recursion(14) = recursion(13) = recursion(12) = recursion(11)이 된다.

즉, recursion(16)
= 16 + recursion(15) = 16 + recursion(11)
= 16 + 11 + recursion(10) = 16 + 11 + recursion(6)
= 16 + 11 + 6 + recursion(5)
= 16 + 11 + 6 + recursion(4)
여기에서 recursion(4)를 실행하면 recursion 함수 내에서 첫 번째 줄이 실행되어 1을 반환한다.
따라서 최종 결과값은 16 + 11 + 6 + 1 = 27 + 7 = **34**가 된다.

19. 인터넷에서 주로 사용하는 TCP와 UDP에 대한 설명으로 옳지 않은 것은?

① TCP에는 네트워크 상 트래픽이 폭주하는 현상을 방지하는 혼잡 제어(congestion control) 기능이 포함되어 있다.
② UDP는 비신뢰적이고 비연결형인 서비스를 제공한다.
③ TCP는 패킷 손실이 발생했을 때, 패킷 재전송 기능을 수행한다.
④ UDP 패킷 헤더에는 출발지 포트 번호, 목적지 포트 번호, 길이, 체크섬 등의 정보를 담고 있다.
⑤ TCP는 신뢰적인 데이터 전달은 보장하지만, 수신 프로세스에게 데이터가 올바른 순서로 전달되는 것은 보장하지 못한다.

답 ⑤

⑤ TCP를 통해 전송할 경우 순서대로 전송하지만 네트워크 경로상의 문제로 인해 수신측에서는 순서대로 전송받지 못하는 경우가 발생한다. 하지만 이후 TCP는 순서에 맞춰 패킷을 재조립하기 때문에 올바른 순서로 전달되는 것을 보장한다.

<오답 체크> ①③ TCP는 OSI 4계층 전송계층에서 작동하는 신뢰성, 연결 지향형의 프로토콜이다.
전이중 전송방식/양방향성(full-duplex)의 특성을 가지고 있으며, 흐름제어, 혼잡제어 등의 기능을 수행한다.
신뢰성 연결 방식이기 때문에 패킷 손실이 발생할 경우 재전송을 한다.

④ UDP 헤더에도 체크섬은 들어있다. 다만, TCP에서 헤더 체크섬(checksum)은 필수 사항지만 UDP에서 헤더 체크섬은 선택 사항이다.

20. 다음 중 회선 교환 방식(circuit switching)의 특성으로 옳지 않은 것은?

① 데이터를 주고 받기 전에 종단 간 연결(end to end connnection) 과정이 필요하다.
② 연결이 수립되면, 해당 회선을 독점적으로 사용한다.
③ 회선 교환 방식의 대표적인 예로는 전화망이 있다.
④ 패킷 교환 방식(packet switching)에 비해 더 많은 동시 접속자를 수용할 수 있다.
⑤ 패킷 교환 방식에 비해 안정적인 전송속도를 보장할 수 있다.

답 ④

④ 회선 교환 방식에서는 접속이 수립되면, 송수신 양측 시스템이 통신회선을 독점하게 된다. 따라서 동시 접속자를 수용할 수 없다.
그리고 잘 보면 ④번의 내용은 위의 ②번과 모순되는 내용임을 알 수 있다.