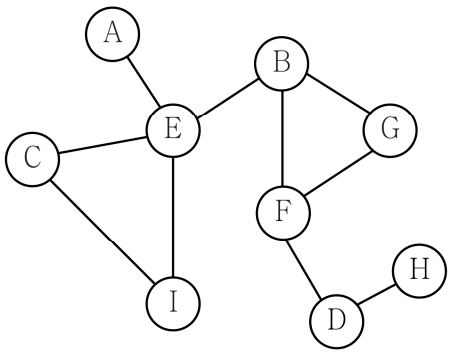
자료구조론

1. 다음 C 코드의 시간 복잡도(time complexity)는?

```
void func(int n)
{
  int i, j, k = 0;
  for(i = 0; i < (n + 200); i++)
    for(j = 0; j < 300; j++)
      printf("k = %d\n", k++);
}
```

① $O(n)$
② $O(n^2)$
③ $O(\log_2 n)$
④ $O(n\log_2 n)$

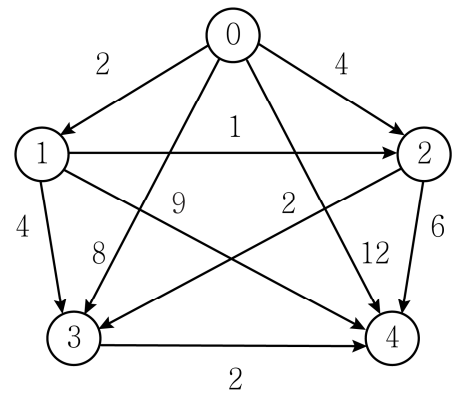
2. 다음 그래프(graph)에 대하여 깊이 우선 탐색(DFS, depth first search)을 하려고 한다. 시작 정점(vertex)이 B일 때, 정점 I는 몇 번째에 방문하는가? (단, B가 첫 번째 방문 정점이고, 탐색 시 인접한 정점들은 알파벳 오름차순으로 방문한다)



① 4
② 5
③ 8
④ 9

3. 이중 연결 리스트(doubly linked list)에 대한 설명으로 옳지 않은 것은?
① 이전 노드와 이후 노드에 대한 링크(link)를 가지고 있어 양방향 탐색이 가능하다.
② 단순 연결 리스트(singly linked list)에 비해 기억 장소가 추가로 소요된다.
③ 이진 트리(binary tree)에 활용이 가능하다.
④ 다항식(polynomial), 희소 행렬(sparse matrix)을 표현할 수 없다.

4. 다음은 정점 간의 거리를 표시한 방향 그래프(directed graph)이다. 시작 정점이 0일 때, 정점 3과 정점 4까지 최단 경로의 거리를 바르게 연결한 것은?



	정점 3	정점 4
①	5	7
②	5	8
③	8	11
④	8	12

5. 어떤 트리 3개에 대한 모든 노드 개수의 합이 100이면 간선(edge) 개수의 합은?
① 94
② 95
③ 96
④ 97

6. 다음 하노이 탑 문제를 해결하는 파이썬(Python) 코드의 시간 복잡도는? (단, n은 원판의 개수이고 first, second, third는 기둥 번호이다)

```
def hanoiT(n, first, second, third):
  if n == 1:
    print("원판1 : %s 에서 %s"%(first, third))
  else:
    hanoiT(n - 1, first, third, second)
    print("원판%d : %s 에서 %s"%(n, first, third))
    hanoiT(n - 1, second, first, third)
```

① $O(n)$
② $O(n^2)$
③ $O(2^n)$
④ $O(\log_2 n)$

7. 어떤 이진 트리의 전위 순회(preorder traversal)와 중위 순회(inorder traversal)한 순서가 다음과 같을 때, 이 트리의 단말 노드만을 모두 나열한 것은?

> ○ 전위 순회: A, B, D, E, C, F, G, H, I
> ○ 중위 순회: E, D, B, A, G, F, H, C, I

① A, D, E
② B, C, D, F
③ D, E, F, I
④ E, G, H, I

8. 다음 비어 있는 크기가 11인 해시 테이블(hash table)에 입력키가 4, 6, 15, 26, 17 순서로 저장되었다. 입력키 26이 저장된 해시 테이블의 주소는? (단, 해시 함수는 h(k) = k mod 11이고, 충돌 해결은 선형 조사법(linear probing)을 사용한다)

해시 테이블 주소	0	1	2	3	4	5	6	7	8	9	10
해시 테이블											

① 1
② 4
③ 6
④ 7

9. 다음은 유클리드 알고리즘(euclidean algorithm)을 사용하여 두 정수의 최대공약수(greatest common divisor)를 구하는 파이썬 프로그램이다. gcd(12, 21)를 실행한 후 count 값과 (가)에 들어갈 내용을 바르게 연결한 것은?

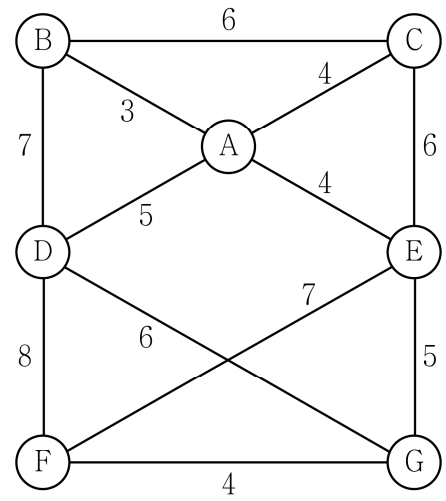
```
def gcd(m, n):
    global count
    count = count + 1
    if m < n:
        temp = m
        m = n
        n = temp
    if m % n == 0:
        return n
    else:
        return gcd(  (가)  )

count = 0
gcd(12, 21)
```

	count	(가)
①	2	m, m % n
②	2	n, n % m
③	3	m, n % m
④	3	n, m % n

10. 배열(array)로 표현한 이진 트리 중에서 최소 힙(minimum heap)에 해당하지 않는 것은?

①	1	3	13	7	9	10	20	17	25
②	1	3	13	9	7	20	17	25	10
③	1	3	17	7	13	25	20	9	10
④	1	9	3	17	10	13	7	20	25

11. 다음 그래프에서 크루스칼 알고리즘(Kruskal algorithm)을 사용하여 만든 최소 비용 신장 트리(minimum cost spanning tree)에 대한 모든 간선의 가중치 합은?



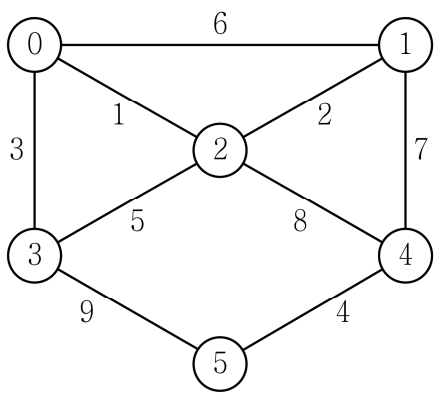
① 21
② 25
③ 31
④ 33

12. 다음 treetest() 함수로 만들어지는 트리의 유형은?

```
typedef struct treeNode {
    int data;
    struct treeNode *left, *right;
} treeNode;

void treetest() {
    treeNode n1 = { 12, NULL, NULL };
    treeNode n2 = { 33, NULL, NULL };
    treeNode n3 = { 21, &n1, &n2 };
    treeNode n4 = { 74, NULL, NULL };
    treeNode n5 = { 67, NULL, &n4 };
    treeNode n6 = { 59, NULL, &n5 };
    treeNode n7 = { 46, &n3, &n6 };
    treeNode* root = &n7;
}
```

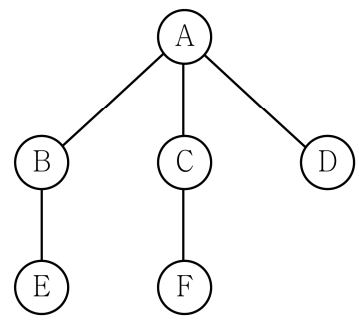
① 스레드 이진 트리(threaded binary tree)
② 완전 이진 트리(complete binary tree)
③ 이진 탐색 트리(binary search tree)
④ 포화 이진 트리(full binary tree)

13. 다음 그래프에서 프림 알고리즘(Prim algorithm)을 사용하여 최소 비용 신장 트리를 만들 때, 트리에 네 번째로 포함되는 간선은? (단, 시작 정점은 0이다)



① (0, 1)
② (1, 4)
③ (2, 3)
④ (4, 5)

14. 다음 그래프에서 정점 A부터 너비 우선 탐색(BFS, breadth first search)을 사용하여 나올 수 있는 결과로 옳지 않은 것은?



① A B C D E F
② A C B D F E
③ A C F B E D
④ A D C B F E

15. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>

void func(int *array, int size) {
  int *p = array;
  *p = 100;
  *(p + 2) = 100;
  p = p + 1;
}

int main(int argc, char *argv[]) {
  int array[5] = { 2, 4, 6, 8, 10 };
  func(array, 5);
  for(int i = 0; i < 5; i++)
    printf("%d ", array[i]);
  printf("\n");
  return 0;
 }
```

① 100 4 6 8 100
② 100 4 6 100 10
③ 100 4 100 8 10
④ 100 100 68 8 10

16. 다음 인접 행렬(adjacency matrix)로 표현된 그래프에 대해 위상 정렬(topological sort)을 수행한 후, 위상 순서(topological order)에서 마지막에 위치하는 정점은? (단, 위상 정렬 과정에서 선택 가능한 정점이 여러 개인 경우 가장 작은 숫자의 정점을 선택한다)

	0	1	2	3	4	5	6	7
0		1		1			1	
1			1					
2								
3		1						
4	1		1			1		
5							1	
6				1				1
7				1				

← 도착 정점

↑ 시작 정점

① 1
② 2
③ 3
④ 6

17. 다음 C 코드에 10개의 노드로 이루어진 이중 연결 리스트를 입력하여 수행한 후, 이중 연결 리스트에 남아 있는 노드의 개수는? (단, 10개 노드 입력 후 수행 시 start는 첫 번째 노드를 가리킨다)

```
typedef struct node {
  int key;
  struct node *prev, *next;
} node;

void foobar(node *start) {
    node *current = start;
    int iseven = 0;
    while(current != NULL) {
      if(iseven == 1) {
        current->prev->next = current->next;
        if(current->next != NULL)
          current->next->prev = current->prev;
      }
      current = current->next;
      iseven = 1 - iseven;
    }
}
```

① 5
② 6
③ 8
④ 10

18. 다음 미로 찾기 문제를 스택(stack)을 이용하여 해결하고자 한다. 출발점에서 시작하여 도착점까지 가기 위해 필요한 push 연산 횟수와 pop 연산 횟수는? (단, 출발점, 도착점 좌표에 대해서는 push 연산 횟수에서 제외한다)

출발 →

0	1	0	1
1	0	1	0
0	0	1	1
1	0	0	0

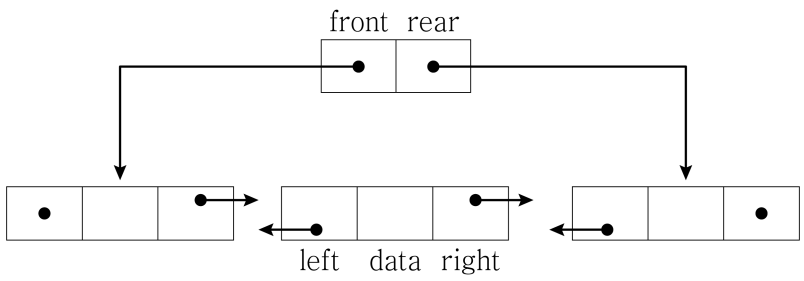
→ 도착

N
W E
S

○ 출발점은 (0, 0), 도착점은 (3, 3)이다.
○ 출발점에서 현재 좌표까지 이동 경로는 push 연산으로 현재 좌표를 스택에 저장하고, 직전 좌표로 돌아가는 경로는 pop 연산으로 현재 좌표를 스택에서 삭제하는 방법으로 만들어진다.
○ 미로의 크기는 4×4 2차원 배열로 구성되었고, 1은 장벽, 0은 통과할 수 있는 길이다. 4×4 2차원 배열 외부는 모두 장벽으로 간주하여 이 배열을 벗어날 수 없다고 가정한다.
○ 각 좌표에서 이전에 이동했던 좌표를 제외하고, N(북)에서 NE(북동), E(동), SE(남동), S(남), SW(남서), W(서), NW(북서) 순서로 검색하여 가장 먼저 찾은 이동 가능한 좌표로 이동한다. 주위 좌표가 모두 벽이거나 방문했던 곳일 경우 직전 좌표로 돌아간다.

	push	pop
①	3	0
②	4	0
③	5	2
④	6	2

19. 다음은 이중 연결 리스트를 이용하여 덱(deque)을 구현한 파이썬 코드이다. (가) ~ (라)에 들어갈 내용을 A ~ F에서 바르게 연결한 것은?




```
class DQNode:
    def __init__(self, data, left, right):
        self.data = data
        self.left = left
        self.right = right

class DoublyLinkedDQ:
    def __init__(self):
        self.front = None
        self.rear = None

    def insertFront(self, data):
        node = DQNode(data, None, self.front)
        if self.front == None:
            self.front = node
            self.rear = self.front
        else:
            (가)
            (나)

    def deleteRear(self):
        if self.front != None:
            data = self.rear.data
            (다)
            if self.rear == None:
                (라)
            else:
                self.rear.right = None
            return data
```

A. self.front = node
B. self.front = self.rear
C. self.front.left = node
D. self.front.right = node
E. self.rear = self.rear.left
F. self.rear = self.rear.right

	(가)	(나)	(다)	(라)
①	A	C	F	A
②	C	A	E	B
③	C	A	F	D
④	D	C	E	C

20. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
typedef struct stack {
  int data[10]; int top;
} stack;
stack s1, s2;

void push(stack *ss, int i) {
  ss->top = ss->top + 1;
  ss->data[ss->top] = i;
}

int pop(stack *ss) {
  return ss->data[ss->top--];
}

int size(stack *ss) {
  return ((ss->top) + 1);
}

void push2(int n) {
  push(&s1, n);
}

int pop2() {
  if(size(&s2) == 0) {
    int s = size(&s1);
    for(int i = 0; i < s; i++)
      push(&s2, pop(&s1));
  }
  return pop(&s2);
}

int main(int argc, char **argv) {
  s1.top = s2.top = -1;
  push2(1);  push2(2);  push2(3);
  printf("%d  ", pop2());
  printf("%d  ", pop2());
  push2(4);  push2(5);
  printf("%d  ", pop2());
  printf("%d  ", pop2());
  printf("%d  ", pop2());
  return 0;
}
```

① 1 2 3 4 5
② 2 1 5 4 3
③ 3 2 5 4 1
④ 5 4 3 2 1

21. 다음 데이터를 비어 있는 AVL(Adelson-Velskii, Landis) 트리에 순서대로 삽입할 때, 완전 이진 트리에 해당하는 것만을 모두 고른 것은?

ㄱ. 1, 2, 4, 3, 6, 5
ㄴ. 2, 1, 5, 4, 6, 3
ㄷ. 3, 1, 4, 2, 5, 6
ㄹ. 4, 1, 2, 3, 6, 5

① ㄱ, ㄴ
② ㄱ, ㄹ
③ ㄴ, ㄷ
④ ㄷ, ㄹ

22. 다음 데이터를 비어 있는 2-3 트리에 순서대로 삽입하였다. 8을 삭제한 후, 트리에서 레벨 1의 값을 나열하면? (단, 루트 노드의 레벨은 0이다)

20, 14, 35, 17, 23, 10, 5, 8, 27, 31

① 10, 20
② 14, 27
③ 8, 20, 27
④ 17, 20, 27

23. 숫자 분석법(digit analysis)은 모든 키 값이 미리 알려져 있을 때 유용한 해시 함수이다. 키 집합이 다음과 같을 때, 숫자 분석법을 이용하여 십진수 두 자리의 주소를 생성하고자 한다. 두 자리 주소에 들어갈 자릿수로 가장 적절한 것은?

0001	1006	2011	3016	4021	5026	6031	7036	8041	9046
0002	1007	2012	3017	4022	5027	6032	7037	8042	9047
0003	1008	2013	3018	4023	5028	6033	7038	8043	9048
0004	1009	2014	3019	4024	5029	6034	7039	8044	9049
0005	1010	2015	3020	4025	5030	6035	7040	8045	9050

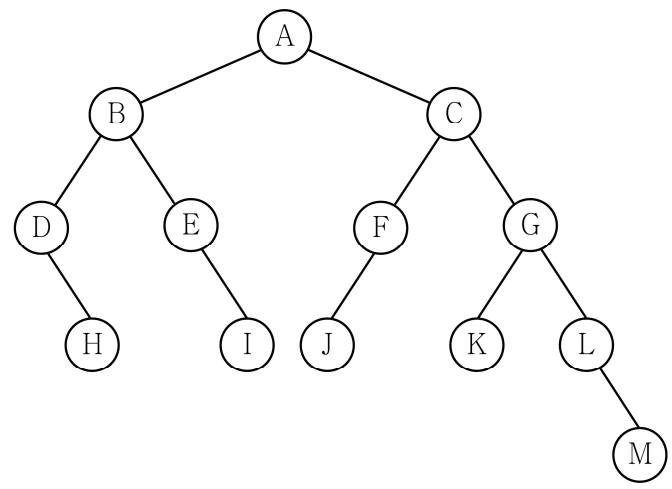
① 100의 자릿수, 1의 자릿수
② 100의 자릿수, 10의 자릿수
③ 1000의 자릿수, 1의 자릿수
④ 1000의 자릿수, 10의 자릿수

24. 다음 파이썬 프로그램의 실행 결과는?

```
import queue


values=queue.Queue()
for i in range(20):
   if i % 3 == 0:
      values.put(i)
   elif i % 4 == 0:
      values.get()
   elif i % 5 == 0:
      values.get()
print(values.get(), values.get())
```

① 15
② 18
③ 15 18
④ 12 15 18

25. 다음 트리는 노드에서 키와 값이 무엇인지 알 수는 없지만 이진 탐색 트리라고 가정했을 때, 이에 대한 설명으로 옳지 않은 것은? (단, 루트 노드만 있는 경우에 트리 높이는 0이다)



① 트리의 높이는 4이다.
② AVL 트리에 해당한다.
③ C 노드의 중위 후속자(inorder successor)는 M 노드이다.
④ 높이를 증가시키지 않고 트리에 추가할 수 있는 최대 노드 수는 18개이다.