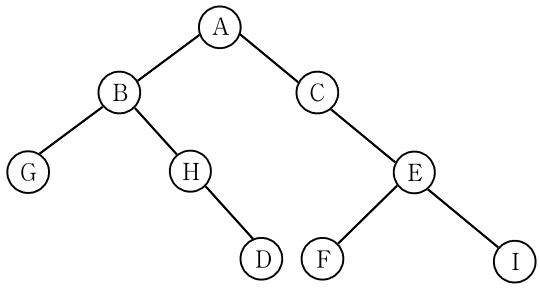
자료구조론

문 1. 다음 C 언어 함수를 사용하여 function(9)를 수행했을 때, 반환되는 값은?

```
int function(int n)
{
  if (n <= 1) return 1;
  if (n % 2 == 0)
    return n * function(n-1);
  else
    return n * function(n-3);
}
```

① 60 ② 84
③ 540 ④ 672

문 2. 다음 이진 트리(binary tree)의 순회 방법으로 옳지 않은 것은?



① 전위(preorder) 순회 : A B G H D C E F I
② 중위(inorder) 순회 : G B H D A C F E I
③ 후위(postorder) 순회 : G D H B F E I C A
④ 레벨 순서(level order) 순회 : A B C G H E D F I

문 3. 다음과 같이 인접 행렬로 표현된 가중 그래프(weighted graph)에 대하여 최소 비용 신장 트리(minimum cost spanning tree)를 구성했을 때, 최소 비용은?

	[0]	[1]	[2]	[3]
[0]	0	5	1	2
[1]	5	0	∞	4
[2]	1	∞	0	3
[3]	2	4	3	0

① 9 ② 8
③ 7 ④ 6

문 4. 78개의 간선(edge)으로 이루어진 이진 트리(binary tree)가 가질 수 있는 최대 높이와 최소 높이의 차이는? (단, 루트 노드만 존재하는 이진 트리의 높이는 1이다)

① 71 ② 72
③ 73 ④ 74

문 5. 1과 1000 사이의 정수로 구성된 이진 탐색 트리(binary search tree)에서 숫자 629를 탐색한다고 할 때, 옳지 않은 비교 순서는?

① 19, 929, 172, 891, 224, 735, 405, 674, 537, 629
② 876, 807, 24, 186, 314, 717, 690, 428, 512, 629
③ 29, 829, 815, 137, 369, 504, 805, 589, 692, 629
④ 926, 87, 813, 318, 572, 826, 635, 598, 580, 629

문 6. 단순 연결 리스트(singly linked list)의 노드를 나타내는 구조체 node에 다음 노드를 가리키는 포인터 link가 있다고 하자. 2개의 노드로 구성된 단순 연결 리스트에서 첫 번째 노드와 두 번째 노드의 사이에 새로운 노드를 삽입하면서 리스트의 순서를 역순으로 변환하고자 한다. p에는 리스트의 시작 주소가 저장되어 있으며, new에는 새로운 노드의 주소가 저장되어 있다고 가정할 때, 명령문의 실행 순서로 옳은 것은? (단, 역순 변환 후 p에는 리스트의 새로운 시작 주소가 저장되어 있어야 한다)

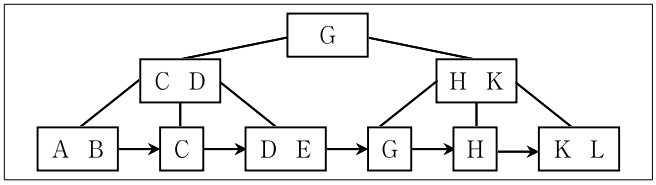
① new→link = p; p = p→link; p→link = new;
new→link→link = NULL;
② new→link = p; p→link = new; p = p→link;
new→link→link = NULL;
③ p = p→link; p→link = new; new→link = p;
new→link→link = NULL;
④ p→link = new; p = p→link; new→link = p;
new→link→link = NULL;

문 7. 다음 자료에서 첫 행은 오름차순으로 정렬하기 위해 주어진 입력 값이고 마지막 행은 정렬이 완료된 후의 결과이다. 중간의 행들은 합병정렬(merge sort), 삽입정렬(insertion sort), 선택정렬(selection sort), 퀵정렬(quick sort) 알고리즘을 통하여 얻을 수 있는 중간 과정을 보여준다. 각 행에 사용된 정렬 알고리즘을 바르게 나열한 것은? (단, 퀵정렬에서 피봇은 가장 오른쪽에 있는 원소를 사용한다)

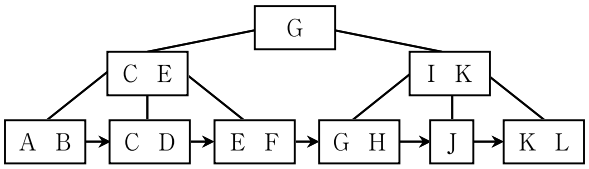
입력	21	41	15	25	21	37	45	9	23	18	36	4	11
㉠	9	15	21	21	25	37	41	45	23	18	36	4	11
㉡	4	9	11	15	18	37	45	41	23	21	36	21	25
㉢	15	21	21	25	37	41	45	9	18	23	4	11	36
㉣	4	9	11	25	21	37	45	41	23	18	36	21	15
결과	4	9	11	15	18	21	21	23	25	36	37	41	45

㉠	㉡	㉢	㉣
① 삽입정렬	선택정렬	합병정렬	퀵정렬
② 삽입정렬	퀵정렬	합병정렬	선택정렬
③ 합병정렬	선택정렬	삽입정렬	퀵정렬
④ 합병정렬	퀵정렬	선택정렬	삽입정렬

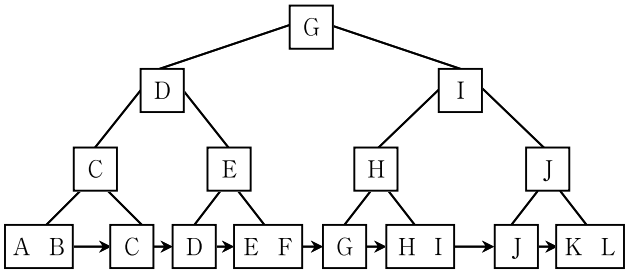
문 8. 다음은 차수가 3인 B+ 트리(tree)의 초기 상태이다. 이 트리에 색인키 'F', 'I', 'J'를 순서대로 삽입한 후의 모습은? (단, 단말 노드는 최대 2개의 원소를 포함할 수 있으며, 단말 노드의 분할 시 중간 위치의 값은 오른쪽 노드에 포함되도록 한다)



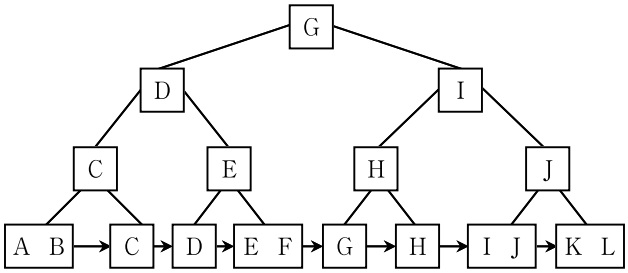
①



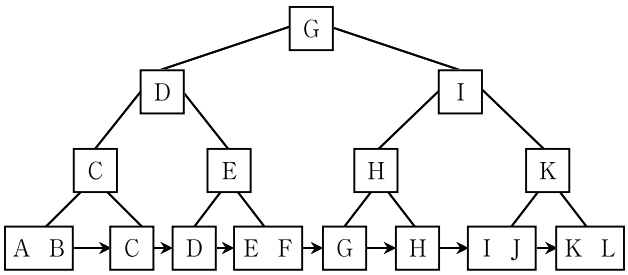
②



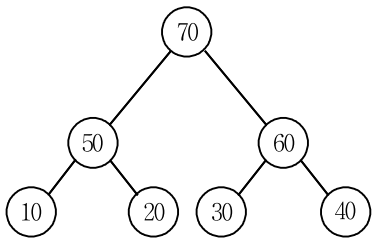
③



④



문 9. 그림과 같은 최대 힙(max heap)을 구성하기 위한 삽입 순서로 옳은 것은?



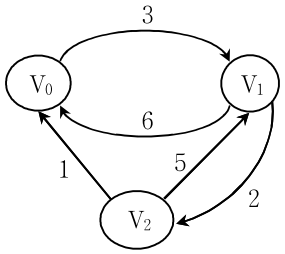
① 50, 60, 30, 10, 20, 70, 40

② 10, 20, 40, 30, 60, 50, 70

③ 10, 60, 50, 70, 20, 30, 40

④ 50, 30, 40, 70, 60, 10, 20

문 10. 다음 그래프에 대한 모든 쌍의 최단 경로를 구하기 위해 최단 거리 행렬 D^{-1}, D^0, D^1, D^2를 차례대로 구한다고 했을 때, D^1의 계산 결과는?



①

	[0]	[1]	[2]
[0]	0	3	∞
[1]	6	0	2
[2]	1	4	0

②

	[0]	[1]	[2]
[0]	0	3	5
[1]	6	0	2
[2]	1	4	0

③

	[0]	[1]	[2]
[0]	0	3	5
[1]	3	0	2
[2]	1	4	0

④

	[0]	[1]	[2]
[0]	0	3	∞
[1]	6	0	2
[2]	1	5	0

문 11. 다음은 연결 스택(linked stack)에서 새로운 노드를 삽입하는 C 언어 프로그램이다. ㉠, ㉡에 들어갈 내용으로 옳게 짝지은 것은? (단, top은 스택의 첫 번째 노드를 가리키는 포인터이며, top이 NULL이면 공백 스택이다)

```
typedef struct node {
  int key;
  struct node *link;
} StackNode;

StackNode *top = NULL;

void push(int data) {
  StackNode *tmp = (struct node*)malloc(sizeof(StackNode));
  if (tmp == NULL) {
    printf("Memory allocation is failed.\n");
    exit(1);
  }
  tmp→key = data;
  if (top == NULL) {
    tmp→link = NULL;
    top = tmp;
  }
  else {
    [ ㉠ ];
    [ ㉡ ];
  }
}
```

	㉠	㉡
①	tmp→link = top	top = tmp
②	top→link = tmp	tmp = top→link
③	top→link = tmp	tmp = top
④	tmp→link = top	top = tmp→link

문 12. 다음 키(key)를 갖는 데이터들을 공백 AVL 트리에 차례대로 저장하였다. 생성된 AVL 트리를 T라고 할 때, 다음 설명 중에서 옳은 것은?

12, 14, 9, 16, 19, 11

① T에 키가 25인 노드를 삽입하면 루트 노드가 바뀌게 된다.
② T에서 키가 9인 노드를 삭제하면 T의 높이가 1 감소한다.
③ T에 키가 7인 노드를 삽입하면 T의 높이가 1 증가한다.
④ 키가 9인 노드와 키가 16인 노드는 형제 노드이다.

문 13. 다음 C 언어 함수의 시간 복잡도를 빅오(O) 표기법으로 표현한 것은? (단, $n > 1$)

```
void testing(int n)
{
  int i, j, sum = 0;
  for (i = 0; i < n; i++)
    for (j = n; j > 1; j /= 2)
      sum += 1;
}
```

① O(logn)
② O(nlogn)
③ $O(n^2)$
④ O(n)

문 14. 다음은 원형 연결 리스트(circular linked list)의 길이를 구하는 C 언어 프로그램이다. ㉠ ~ ㉢에 들어갈 내용으로 옳게 짝지은 것은?

```
typedef struct node {
  int key;
  struct node *link;
} list_node;
int length(list_node *ptr) {
  list_node *temp;
  int count = 0;
  if ( ㉠ ) {
    temp = ptr;
    do {
      count++;
      ㉡ ;
    } while ( ㉢ );
  }
  return count;
}
```

	㉠	㉡	㉢
①	ptr != NULL	temp = temp→link	temp != ptr
②	ptr == NULL	temp = temp→link	temp == ptr
③	ptr != NULL	ptr = temp→link	temp != ptr
④	ptr == NULL	ptr = temp→link	temp == ptr

문 15. 이중 연결 리스트(doubly linked list)의 노드를 나타내는 구조체 node에 이전 노드를 가리키는 포인터 llink와 다음 노드를 가리키는 포인터 rlink가 있다고 하자. 포인터 new_node가 가리키는 노드를 포인터 p가 가리키는 노드의 왼쪽에 삽입할 때, 문장의 수행 순서를 바르게 나열한 것은?

```
ㄱ. p→llink→rlink = p→rlink;
ㄴ. p→rlink→llink = p→llink;
ㄷ. p→llink→rlink = new_node;
ㄹ. new_node→rlink = p;
ㅁ. p = new_node;
ㅂ. new_node→llink = p→llink;
ㅅ. p→llink = new_node;
```

① ㄱ → ㄴ
② ㅂ → ㅅ → ㄷ → ㄹ
③ ㄷ → ㅂ → ㄹ → ㅁ
④ ㄷ → ㄹ → ㅂ → ㅅ

문 16. 다음은 C 언어를 사용하여 원형 큐(circular queue)의 삽입(enqueue)과 삭제(dequeue) 연산을 구현한 것이다.

```
int Q[8] = {0};
int front = 0;
int rear = 0;

void enqueue(int item)
{
  rear = (rear + 1) % 8;
  if (front == rear) {
    printf("Queue is full.\n");
    return;
  }
  Q[rear] = item;
}

int dequeue()
{
  int item;
  if (front == rear) {
    printf("Queue is empty.\n");
    exit(1);
  }
  else {
    front = (front + 1) % 8;
    item = Q[front];
    Q[front] = 0;
    return item;
  }
}
```

front와 rear의 값이 0인 공백 큐에 6개의 값 1, 3, 2, 4, 8, 6을 차례대로 삽입하고, 여기에서 2개의 값을 삭제한 다음, 다시 3개의 값 5, 7, 9를 차례대로 삽입할 때, 배열 Q의 최종 모습은?

	Q[0]	Q[1]	Q[2]	Q[3]	Q[4]	Q[5]	Q[6]	Q[7]
①	9	0	2	4	8	6	5	7
②	7	9	0	2	4	8	6	5
③	2	4	8	6	5	7	9	0
④	0	7	9	2	4	8	6	5

문 17. 다음 C 프로그램의 출력 값으로 옳은 것은? (단, int 데이터 타입은 4 바이트로 표현되며, 배열 a의 시작 주소는 2293520이다)

```
int main()
{
  int a[3][4] = {{10, 15, 20, 25}, {30, 35, 40, 45},
                 {50, 55, 60, 65}};
  printf("a+1 = %d, ", a+1);
  printf("*(*a+2) = %d, ", *(*a+2));
  printf("*(*(a+1)+2) = %d\n", *(*(a+1)+2));
  return 0;
}
```

① a+1 = 2293521, *(*a+2) = 50, *(*(a+1)+2) = 55

② a+1 = 2293536, *(*a+2) = 20, *(*(a+1)+2) = 40

③ a+1 = 2293521, *(*a+2) = 20, *(*(a+1)+2) = 40

④ a+1 = 2293536, *(*a+2) = 50, *(*(a+1)+2) = 55

문 18. 다음 인접행렬이 표현하는 그래프의 부분 그래프 중에서 간선(edge)이 최소한 1개 이상인 연결 그래프의 개수는?

	[0]	[1]	[2]	[3]
[0]	0	1	1	1
[1]	1	0	1	0
[2]	1	1	0	0
[3]	1	0	0	0

① 12

② 13

③ 14

④ 15

문 19. 기수 정렬(radix sort)을 수행하여 다음 원소들을 오름차순으로 정렬하고자 한다. 각 단계별 정렬 순서로 나타날 수 없는 것은? (단, 정수 K는 $0 \leq K \leq 999$ 범위에 있고 각 자리수를 나타내는 3개의 서브키(K^1, K^2, K^3)로 구성되어 있다고 생각한다. 여기서 K^1는 일의 자리 수, K^2는 십의 자리 수, K^3는 백의 자리 수를 의미하고, $0 \leq K^i \leq 9$이다)

129, 308, 506, 92, 3, 841, 33

① 3, 33, 92, 129, 308, 506, 841

② 841, 92, 3, 33, 506, 308, 129

③ 92, 3, 33, 129, 308, 506, 841

④ 3, 506, 308, 129, 33, 841, 92

문 20. 행렬 $M=(a_{ij})$가 $|i-j|>1$ 이면 0 값을 가질 때, 이러한 행렬을 삼중대각행렬(tridiagonal matrix)이라고 하며, 다음은 6×6 삼중대각행렬의 예를 보여준다. (단, $1 \leq i \leq m$, $1 \leq j \leq n$)

$$\begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} & 0 \\ 0 & 0 & 0 & a_{54} & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} \end{bmatrix}$$

행렬 $M(m\times n)$의 0이 아닌 원소 a_{ij}를 열 우선(column major) 순서로 배열 A에 저장한다고 할 때, 0이 아닌 원소 a_{ij}가 배열 A에 저장되는 위치를 계산하는 수식으로 옳은 것은? (단, 배열 A의 시작주소는 α이고, 행렬 M의 0인 원소는 배열에 저장하지 않는다)

① $\alpha+3 \cdot (j-1)+(i-j)$

② $\alpha+3 \cdot (j-1)+(j-i)$

③ $\alpha+3 \cdot (i-1)+(i-j)$

④ $\alpha+3 \cdot (i-1)+(j-i)$