

## 자료구조론

문 1. 다음 C 언어 함수에 의해 구현된 정렬 방식은?

```
void whatsort(int a[ ], int size)
{
    int i, j, temp;
    for (i = (size-1); i > 0; i--) {
        for (j = 1; j <= i; j++) {
            if (a[j-1] > a[j]) {
                temp = a[j-1];
                a[j-1] = a[j];
                a[j] = temp;
            }
        }
    }
}
```

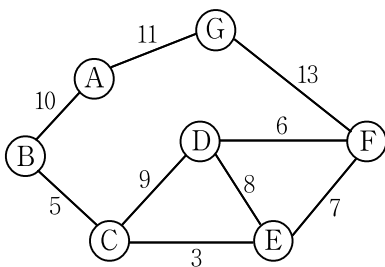
- ① 삽입 정렬(insertion sort)
- ② 선택 정렬(selection sort)
- ③ 힙 정렬(heap sort)
- ④ 버블 정렬(bubble sort)

문 2. 다음 중위 표현식(infix expression)을 후위 표현식(postfix expression)으로 변환한 후, 스택을 이용하여 후위 표현식을 계산하고자 한다. 후위 표현식의 계산 과정에서 스택에 여덟 번째로 삽입(push)되는 값은?

$$(4 + 2) / 3 + (6 / 2 - 1)$$

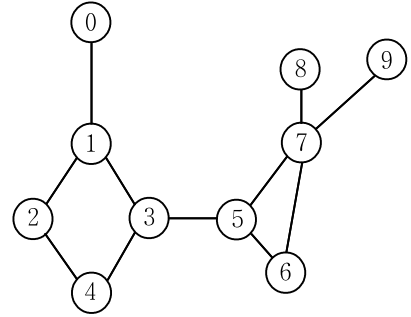
- ① 1
- ② 2
- ③ 3
- ④ 4

문 3. Prim 알고리즘을 사용하여 다음 가중 그래프(weighted graph)의 최소 비용 신장 트리(minimum cost spanning tree)를 구성할 때, 최소 비용과 마지막으로 선택되는 간선은? (단, 시작 정점은 A이다)



- ① 42, (A, G)
- ② 43, (A, G)
- ③ 44, (F, G)
- ④ 45, (F, G)

문 4. 시작 정점이 6일 때, 다음 그래프에 대한 깊이 우선 탐색(DFS: Depth First Search)의 방문 순서는? (단, 인접한 정점들은 오름차순으로 방문한다)



- ① 6, 5, 3, 1, 0, 2, 4, 7, 8, 9
- ② 6, 5, 7, 3, 8, 9, 1, 4, 0, 2
- ③ 6, 5, 3, 4, 2, 1, 0, 7, 8, 9
- ④ 6, 5, 7, 3, 1, 4, 0, 2, 8, 9

문 5. C 언어 함수로 구현된 <정렬 알고리즘>과 크기가 5인 배열 list를 이용하여 sort(list, 5)를 수행하였다. <설명>에서 옳은 것만을 모두 고르면? (단, sort(list, 5) 수행 전에 list 배열에 4, 9, 8, 6, 3이 순서대로 저장되어 있다)

— <정렬 알고리즘> —

```
#define SWAP(x, y, t) ((t)=(x), (x)=(y), (y)=(t))

void sort(int list[ ], int n)
{
    int i, j, least, temp;
    for (i = 0; i < n - 1; i++) {
        least = i;
        for (j = i + 1; j < n; j++)
            if (list[j] < list[least]) least = j;
        SWAP(list[i], list[least], temp);
    }
}
```

— <설 명> —

- ㄱ. sort(list, 5)의 수행이 완료될 때까지 SWAP은 4회 수행된다.
- ㄴ. SWAP이 두 번째 수행된 후 배열 list에는 3, 4, 8, 6, 9가 순서대로 저장되어 있다.
- ㄷ. 정렬 알고리즘의 시간 복잡도를 빅세타(Θ) 표기법으로 표현한 것은 Θ(nlogn)이다.

- ① ㄱ
- ② ㄱ, ㄴ
- ③ ㄱ, ㄷ
- ④ ㄴ, ㄷ

문 6. 공백 AVL 트리에 데이터를 순서대로 삽입하여 AVL 트리를 구성할 때, 나머지와 다른 모습의 AVL 트리가 구성되는 데이터 키(key)들의 순서는?

- ① 3, 4, 6, 5, 7, 8
- ② 5, 4, 7, 3, 6, 8
- ③ 7, 5, 8, 6, 4, 3
- ④ 8, 7, 5, 4, 6, 3

문 7. 다음 키(key) 값을 갖는 9개의 데이터를 순서대로 삽입하여 이진 탐색 트리(binary search tree)를 구성하였다. 구성된 이진 탐색 트리에서 루트 노드를 삭제한 후 구성되는 이진 탐색 트리에 대한 설명으로 옳지 않은 것은?

5, 6, 2, 8, 4, 1, 9, 3, 7

- ① 루트 노드의 키는 4와 6 중 한 개의 값을 가진다.
- ② 중위 순회(inorder traversal)할 경우 키 값이 1, 2, 3, 4, 6, 7, 8, 9인 노드들을 순서대로 방문한다.
- ③ 모든 단말 노드들의 키 값들의 합은 21이다.
- ④ 모든 노드들의 차수(degree)들의 합은 7이다.

문 8. 다음 인접 행렬은 5개 도시 간의 직접 거리를 보여준다. 인천-대구 구간 최단 경로 거리와 광주-대구 구간 최단 경로 거리 사이의 차이 값은? (단,  $\infty$ 는 두 도시 간에 직접적인 연결이 없음을 의미한다)

	대전	광주	서울	인천	대구
대전	0	10	15	25	9
광주	10	0	18	20	$\infty$
서울	15	18	0	3	17
인천	25	20	3	0	$\infty$
대구	9	$\infty$	17	$\infty$	0

- ① 1
- ② 2
- ③ 3
- ④ 4

문 9. 키(key) 집합에 속하는 각 키에 대해 탐색이 일어날 확률이 알려져 있을 때, 최적 이진 탐색 트리(optimal binary search tree)를 구성함으로써 한 번의 탐색에 대한 평균 비교 횟수를 최소화할 수 있다.  $p(x)$ 를 키  $x$ 에 대한 탐색 확률이라고 할 때, 키 집합 {1, 2, 3}에 속하는 각 키에 대한 탐색 확률은 다음과 같다. 키 값이 각각 1, 2, 3인 3개의 데이터를 사용하여 구성한 최적 이진 탐색 트리에서 탐색 1회당 평균 비교 횟수는? (단,  $p(1) + p(2) + p(3) = 1$ 이므로, 1, 2, 3 외의 키를 탐색하는 경우는 없다)

$p(1) = 0.6, p(2) = 0.3, p(3) = 0.1$

- ① 1.4
- ② 1.5
- ③ 1.6
- ④ 1.7

문 10. 다음의 해시 함수들을 사용하는 이중 해싱(double hashing)에서 해시 테이블의 크기는 7이며 0부터 6까지의 인덱스를 가진다.  $h(x)$ 는 첫 번째 조사 위치를 결정하는 기본적인 해시 함수이고  $f(x)$ 는 충돌 발생 시 조사 위치 간격을 결정하는 추가 해시 함수로서,  $i$ 번째 충돌 발생 시 다음 조사 위치를 결정하는 해시 함수는  $h_i(x)$ 가 된다. 공백 해시 테이블에 일련의 키(key) 값 9, 10, 2, 3, 16, 13, 11을 가지는 레코드들을 순서대로 삽입한 후, 해시 테이블의 위치별 키 값을 보여주는 것은? (단, 해시 테이블 버킷(bucket)당 슬롯(slot) 수는 1개이다)

$$h(x) = x \bmod 7$$

$$f(x) = 5 - (x \bmod 5)$$

$$h_i(x) = (h(x) + i \times f(x)) \bmod 7$$

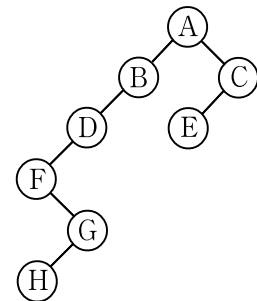
	[0]	[1]	[2]	[3]	[4]	[5]	[6]
①	13	16	9	10	11	2	3
②	16	13	9	10	11	2	3
③	3	13	9	10	11	2	16
④	13	11	9	10	2	3	16

문 11. 다음 일련의 스택 연산에 의해 스택에서 삭제되는 자료를 순서대로 바르게 나열한 것은? (단,  $push(x)$ 는 스택에 자료  $x$ 를 삽입하는 연산이고,  $pop()$ 은 스택에서 자료 한 개를 삭제하는 연산이다)

$push(9), pop(), push(5), push(7), push(2), pop(), push(9), pop(), push(4), pop(), pop(), push(6), push(3), pop(), push(2), pop(), pop()$

- ① 9, 5, 7, 2, 9, 4, 6, 3
- ② 9, 2, 9, 4, 7, 3, 2, 6
- ③ 2, 3, 6, 4, 9, 2, 7, 5
- ④ 9, 5, 9, 4, 7, 6, 2, 2

문 12. 다음 이진 트리(binary tree)를 <보기>에서 제시한 방법을 사용하여 1차원 배열로 표현할 때, 필요한 배열의 최소 크기는? (단, 배열의 인덱스(index)는 0부터 시작하며 배열의 크기는 배열 원소의 개수이다)



<보 기>

- 루트 노드는 배열의 인덱스가 1인 위치에 저장된다.
- 배열 인덱스가  $k$ 인 위치에 있는 노드가 자식 노드를 가질 경우, 왼쪽 자식 노드는 인덱스가  $2k$ 인 위치에 있으며 오른쪽 자식 노드는 인덱스가  $2k+1$ 인 위치에 있다.

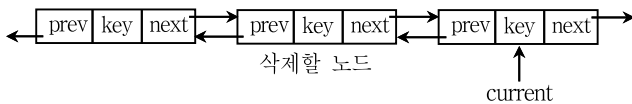
- ① 17
- ② 18
- ③ 34
- ④ 35

문 13. 행 우선(row major) 순서로 저장되는 3차원 배열 `a[3][30][10]`이 있을 때, 첫 번째 원소인 `a[0][0][0]`의 메모리 주소가 2048이면 `a[2][15][2]`의 메모리 주소는? (단, 배열 `a`에서 각 원소의 크기는 4바이트이다)

- ① 2800                                      ② 2804  
③ 5052                                      ④ 5056

문 14. 다음은 C 언어 구조체와 이를 이용한 이중 연결 리스트(doubly linked list)이다. 포인터 `current`가 가리키는 노드의 왼쪽 노드를 삭제할 때 수행할 C 언어 문장을 순서대로 바르게 나열한 것은? (단, 삭제 전에 포인터 `current`가 가리키는 노드의 왼쪽에 최소 2개의 노드가 있다고 가정하며, 삭제 노드에 대한 메모리 할당 해제에는 생략한다)

```
struct Dnode {
    int key;
    struct Dnode *prev;
    struct Dnode *next;
};
```



- ① `current->next = current->next->next;`  
`current->next->prev = current;`  
 ② `current->next->prev = current;`  
`current->next = current->next->next;`  
 ③ `current->prev = current->prev->prev;`  
`current->prev->next = current;`  
 ④ `current->prev->next = current;`  
`current->prev = current->prev->prev;`

문 15. 다음 C 언어 프로그램의 출력 결과는?

```
#include <stdio.h>

int i;

int RecFunc(int n)
{
    i++;
    if (n <= 1)
        return 1;
    else
        return RecFunc(n-1) + RecFunc(n-2);
}

int main()
{
    int result;
    i = 0;
    result = RecFunc(5);
    printf("%d, %d\n", i, result);
}
```

- ① 13, 5                                      ② 13, 8  
③ 15, 5                                      ④ 15, 8

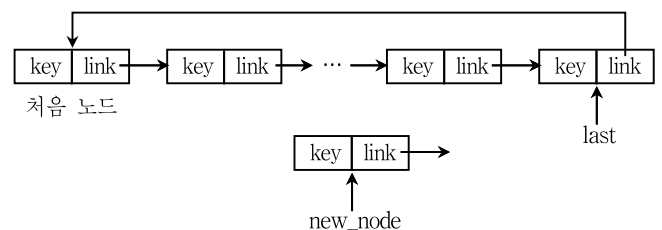
문 16. 다음은 키(key)의 오름차순으로 정렬된 배열을 보여준다. 키 24에 대한 보간 탐색(interpolation search)을 수행할 때, 첫 번째 탐색 위치는? (단, 배열에서 크기가 가장 작은 키와 가장 큰 키의 값은 미리 주어지고, 위치의 차이는 키 값의 차이에 비례한다는 가정 하에 탐색 위치를 계산하며, 소수점 이하는 반올림한다)

위치	0	1	2	3	4	5	6	7	8	9	10	11	12	13
키	3	10	15	19	24	30	40	65	70	74	80	82	87	93

- ① 2  
② 3  
③ 4  
④ 5

문 17. 다음은 C 언어 구조체와 이를 이용한 원형 연결 리스트(circular linked list)이다. 원형 연결 리스트의 맨 마지막에 포인터 `new_node`가 가리키는 노드를 삽입할 때 수행할 C 언어 문장을 순서대로 바르게 나열한 것은? (단, `new_node`의 삽입 전에 원형 연결 리스트는 공백 리스트가 아니고, `last`는 원형 연결 리스트의 마지막 노드를 가리키는 포인터다)

```
struct Cnode {
    int key;
    struct Cnode *link;
};
```



- ① `new_node->link = last->link;`  
`last = new_node;`  
`last->link = new_node;`  
 ② `new_node->link = last->link;`  
`last->link = new_node;`  
`last = new_node;`  
 ③ `last->link = new_node;`  
`new_node->link = last->link;`  
`last = new_node;`  
 ④ `last = new_node;`  
`last->link = new_node;`  
`new_node->link = last->link;`

문 18. 다음 5×5 희소 행렬(sparse matrix)을 구조체 배열로 표현하기 위해 <규칙>의 표현 규칙을 사용하여 <코드>의 C 언어 코드를 작성하였다. <코드>의 ㉠에 들어갈 내용은? (단, 행렬의 행 번호와 열 번호는 0부터 시작한다)

$$\begin{bmatrix} 0 & 5 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

————— <코 드> —————

```
typedef struct {
    int row;
    int col;
    int val;
} term;

term A[ ] = ㉠;
```

————— <규 칙> —————

- 첫 번째 구조체인 A[0]의 row, col, val에는 희소 행렬의 행 개수, 열 개수, 값이 0이 아닌 원소 개수를 순서대로 저장한다.
- 두 번째부터의 나머지 구조체는 값이 0이 아닌 희소 행렬 원소를 표현하기 위해 사용되며, 각 구조체의 row, col, val에 해당 원소의 행 번호, 열 번호, 값을 순서대로 저장한다.
- 값이 0이 아닌 희소 행렬 원소 정보는 행 번호의 오름차순으로 구조체 배열 A에 저장되며, 행 번호가 같은 경우에는 열 번호의 오름차순으로 저장된다.

- ① {{4, 4, 4}, {0, 1, 5}, {0, 2, 2}, {1, 3, 1}, {3, 0, 3}}
- ② {{4, 4, 4}, {3, 0, 3}, {0, 1, 5}, {0, 2, 2}, {1, 3, 1}}
- ③ {{5, 5, 4}, {0, 1, 5}, {0, 2, 2}, {1, 3, 1}, {3, 0, 3}}
- ④ {{5, 5, 4}, {3, 0, 3}, {0, 1, 5}, {0, 2, 2}, {1, 3, 1}}

문 19. 정점이 4개인 무방향(undirected) 완전 그래프(complete graph)에서 만들어질 수 있는 신장 트리(spanning tree)의 총 개수는?

- ① 12
- ② 14
- ③ 16
- ④ 18

문 20. 일련의 키(key) 값 2, 1, 8, 9, 7, 3, 6을 가지는 7개의 데이터를 순서대로 삽입하여 레드-블랙 트리(red-black tree)를 구성하였다. 구성된 레드-블랙 트리를 후위 순회(postorder traversal)할 경우 방문 노드들의 키 값을 방문 순서대로 바르게 나열한 것은?

- ① 1, 3, 6, 2, 9, 8, 7
- ② 1, 3, 7, 6, 9, 8, 2
- ③ 1, 6, 3, 2, 9, 8, 7
- ④ 1, 6, 3, 7, 9, 8, 2