

자료구조론

1. 55개 간선(edge)을 갖는 무방향 완전 그래프의 정점(vertex) 개수는?

- ① 8
② 9
③ 10
④ 11

2. 초기 데이터에 대하여 오름차순 정렬을 2번 연산 수행한 후의 결과가 다음과 같을 때, 사용한 정렬 방법은?

- 초기 데이터: 21, 3, 18, 33, 9, 15
○ 2번 연산 수행 결과: 3, 18, 21, 33, 9, 15

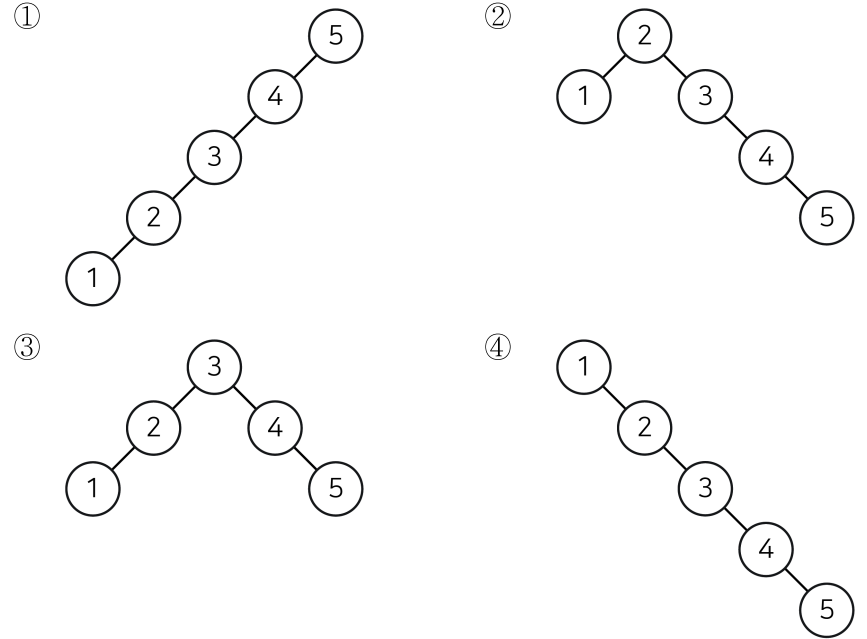
- ① 삽입 정렬
② 선택 정렬
③ 버블 정렬
④ 기수 정렬

3. 다음 C 코드로 표현한 알고리즘의 시간 복잡도는?

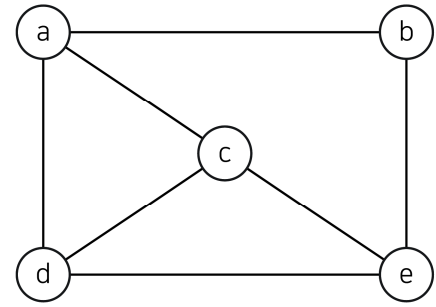
```
Algorithm(int n) {
    int i, j, k;
    int sum = 0;
    for(i = 0; i < 1000; i++)
        for(j = 1; j < (n - 500); j++)
            for(k = (n - 100); k < (n + 100); k++)
                sum = sum + i + j + k;
}
```

- ① $O(1)$
② $O(n)$
③ $O(n^2)$
④ $O(n^3)$

4. 이진 트리(binary tree) 중에서 전위 순회(preorder traversal)와 중위 순회(inorder traversal)한 결과가 동일한 것은?



5. 다음 그래프의 인접 행렬이 아래와 같을 때, (가) ~ (사)에 들어갈 값을 모두 더한 결과는?



	a	b	c	d	e
a	0	(가)	1	1	0
b	(나)	0	0	0	(다)
c	1	0	0	(라)	1
d	1	0	1	0	1
e	0	(마)	1	(바)	(사)

- ① 4
② 5
③ 6
④ 7

6. 다음 C 코드에서 `power(2.0, 20, 0)`를 호출할 때, 변수 `c`의 마지막 출력값은?

```
double power(double x, int n, int c)
{
    printf("c : %d\n", c);
    if(n == 0) return 1;
    else if(n % 2 == 0) {
        c++;
        return power(x * x, n / 2, c);
    }
    else {
        c++;
        return x * power(x * x, (n - 1) / 2, c);
    }
}
```

- ① 5
- ② 6
- ③ 7
- ④ 8

7. 다음 단정도(32비트) IEEE 754 표준 방식으로 표현된 2진수를 10진수 형식으로 옳게 변환한 것은?

[illegible]

- ① 10.5
- ② -10.75
- ③ -20.5
- ④ -20.75

8. 다음 선형 리스트에서 연산 1과 연산 2가 수행된 후 리스트 상태와 두 연산에 사용된 원소의 총이동 횟수를 바르게 연결한 것은?

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
A	B	C	D	E	F	G	

연산 1: 인덱스 1번에 원소 H 삽입
연산 2: 인덱스 3번 원소 삭제

리스트 상태

총이동 횟수

- ①

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
A	H	C		E	F	G	

3

②

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
A	H	C	E	F	G		

3

③

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
A	H	B	D	E	F	G	

10

④

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
A	H	B		D	E	F	G

10

9. 다음은 힙 정렬(heap sort) 알고리즘을 구현한 C 코드이다. 리스트 L이 (61, 11, 59, 15, 48, 19)로 주어졌을 때, for 문에 관한 설명으로 옳은 것만을 모두 고르면? (단, 리스트 L을 배열로 표현할 때 색인 1부터 저장하고, n은 리스트의 크기를 나타낸다)

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

void heap_sort(int L[], int n){
    int i, temp;
    for(i = n/2; i >= 1; i--) ..... (7)
        adjust(L, i, n);
    for(i = n-1; i >= 1; i--) { ..... (4)
        SWAP(L[1], L[i+1], temp);
        adjust(L, 1, i);
    }
}

void adjust(int L[], int root, int n) {
    int next = 2 * root;
    int X = L[root];
    while(next <= n) {
        if(next < n && L[next] < L[next + 1])
            next++;
        if(X > L[next]) break;
        L[next / 2] = L[next];
        next *= 2;
    }
    L[next / 2] = X;
}
```

7. (가) for 문을 수행한 후 리스트 L에는 (61, 48, 59, 15, 19, 11)이 저장된다.

ㄴ. (나) for 문을 첫 1회 수행한 후 리스트 L에는 (59, 48, 19, 15, 11, 61)이 저장된다.

ㄷ. (나) for 문의 시간 복잡도는 $O(n \log n)$ 이다.

- ① $\neg, \perp$
- ② $\neg, \sqsubset$
- ③ $\perp, \sqsubset$
- ④ $\neg, \perp, \sqsubset$

10. 다음 8개의 데이터를 순서대로 삽입하여 AVL 트리를 구성하였을 때, 이에 대한 설명으로 옳은 것은?

19, 7, 10, 15, 25, 32, 17, 5

- ① 루트 노드의 번호는 17이다.
- ② 중위 순회했을 때 네 번째 방문하는 노드의 번호는 15이다.
- ③ 전위 순회했을 때 세 번째 방문하는 노드의 번호는 10이다.
- ④ 후위 순회했을 때 다섯 번째 방문하는 노드의 번호는 32이다.

11. 다음 원형 큐에 대해 1부터 7까지 연산을 차례로 수행했을 때, 수행이 완료된 후 큐의 상태는? (단, 현재 상태는 front = 2, rear = 3이다)

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

연산 1: 원소 B 삽입
 연산 2: 원소 한 개 삭제
 연산 3: 원소 C 삽입
 연산 4: 원소 D 삽입
 연산 5: 원소 두 개 삭제
 연산 6: 원소 E 삽입
 연산 7: 원소 F 삽입

①

[0]	[1]	[2]	[3]	[4]	[5]
			A	E	F

②

[0]	[1]	[2]	[3]	[4]	[5]
D				F	E

③

[0]	[1]	[2]	[3]	[4]	[5]
	F	E	A		

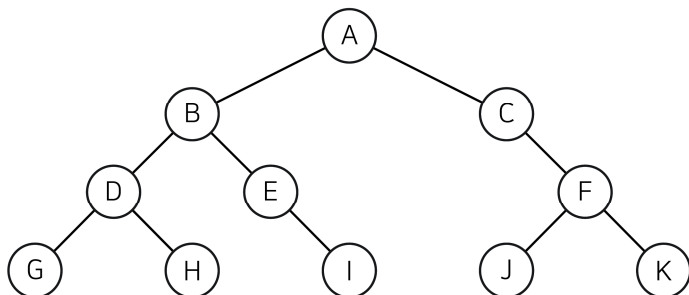
④

[0]	[1]	[2]	[3]	[4]	[5]
D	E	F			

12. 비어 있는 초기 상태의 이진 트리에 다섯 개의 키(1, 2, 3, 4, 5)를 순서대로 하나씩 삽입하면서 레드-블랙 트리를 만든 후, 중위 순회한 결과는?

- ① 1, 2, 3, 4, 5
 ② 1, 3, 5, 4, 2
 ③ 2, 1, 3, 5, 4
 ④ 2, 1, 4, 3, 5

13. 다음 이진 트리에 대해 전위, 중위, 후위 순회(postorder traversal)를 수행할 때, 각 순회 5번째 노드를 바르게 연결한 것은?



	전위	중위	후위
①	E	H	E
②	G	E	H
③	H	E	E
④	H	B	I

14. 다음 C 코드를 사용하여 배열을 선언하였고 arr[0][0][0]의 주소를 a라고 할 때, arr[3][4][2]의 행 우선 순서 주소는?

char arr[6][5][4];

- ① a + 34
 ② a + 78
 ③ a + 96
 ④ a + 112

15. 다음은 정렬된 정수 배열에 대한 이진 탐색(binary search)을 구현한 C 코드이다. (가) ~ (다)에 들어갈 내용을 바르게 연결한 것은?

```

int BSearch(int data[], int size, int target) {
    int mid, left = 0, right = size - 1;
    while(left <= right) {
        mid = (가);
        if(target == data[mid]) return mid;
        else if(target > data[mid]) left = (나);
        else right = (다);
    }
    return -1;
}
  
```

	(가)	(나)	(다)
①	(left+right) // 2	mid-1	mid+1
②	right-(left-right) // 2	mid	left
③	(left+right) / 2	mid+1	mid-1
④	left+(right-left) / 2	right	mid

16. 다음 파이썬 프로그램의 실행 결과는?

```
class Node:
    def __init__(self, data, next):
        self.data = data
        self.next = next
class SinglyLinkedList:
    def __init__(self):
        self.head = None
    def func1(self, data):
        node = self.head
        self.head = Node(data, node)
    def func2(self, data):
        if self.head is None: self.func1(data)
        else :
            node = self.head
            while node.next is not None: node = node.next
            node.next = Node(data, None)
    def func3(self, pos):
        if pos < 0: return None
        node = self.head
        while pos > 0 and node != None:
            node = node.next
            pos -= 1
        return node
    def showList(self):
        node = self.head
        while node :
            print(node.data, end=' ')
            node = node.next

myList = SinglyLinkedList()
myList.func1(1); myList.func2(2)
myList.func1(3); myList.func2(4)
myList.func2(myList.func3(0).data)
myList.showList()
```

- ① 1 2 3 4
 ② 2 3 4 1
 ③ 3 1 2 4 3
 ④ 3 3 1 2 4

17. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
#include <stdlib.h>

typedef struct ListNode {
    int value;
    struct ListNode* next;
} ListNode;

int func(ListNode* p, int n) {
    int i, sum = 0;
    for(i = 0; i < n; i++) {
        sum += p->value;
        p = p->next;
    }
    return sum;
}

int main() {
    int i, s, SIZE = 5;
    ListNode* p[SIZE];
    for(i = 0; i < SIZE; i++) {
        p[i] = (ListNode*)malloc(sizeof(ListNode));
        p[i]->value = i;
    }
    for(i = 0; i < SIZE-1; i++)
        p[i]->next = p[i+1];
    p[SIZE-1]->next = p[0];
    s = func(p[4], 7);
    printf("%d", s);

    return 0;
}
```

- ① 14
 ② 15
 ③ 16
 ④ 17

18. 인접 행렬로 표현된 방향 그래프 중 위상 정렬이 불가능한 것은?

- ①

0100

0010

0001

0000
- ②

0011

0011

0001

0000
- ③

0111

0011

0001

0000
- ④

0101

0010

1001

0000

19. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>

int func(int key, int low, int high, int* list) {
    list[high+1] = key;
    int i = 0;
    for(i = low; list[i] != key; i++);
    if(i == (high+1)) return -1;
    return i;
}

int main(int argc, char* argv[]) {
    int list[100] = {1, 2, 3, 4, 5, 6, 7};
    printf("%d\n", func(3, 4, 6, list));

    return 0;
}
```

- ① -1
- ② 0
- ③ 3
- ④ 4

20. 다음 수식을 후위 표기법(postfix notation)으로 변환한 후, 스택을 이용하여 계산하려고 한다. 계산 과정에서 스택에 일곱 번째로 삽입(push)하는 값은?

(A - B) * C - (D + E) / F

- ① C
- ② D
- ③ E
- ④ (D + E)

21. 다음 C 프로그램의 출력 결과는?

```
#include <stdio.h>
#include <stdlib.h>

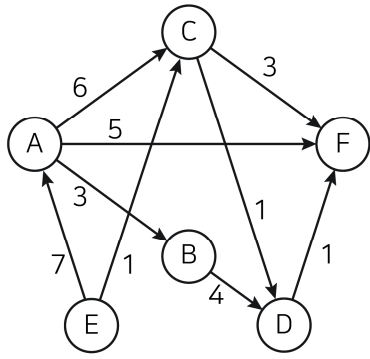
void processArray(int* array, int size) {
    int i = 0;
    for(i = 0; i < size; i++) {
        array[i] += i * 2;
    }
}

void modifyArray(int* array) {
    *(array + 1) = 10;
    *(array + 3) = 20;
}

int main() {
    int i = 0;
    int sum = 0;
    int size = 5;
    int* array = (int*)malloc(size * sizeof(int));
    for(i = 0; i < size; i++) {
        array[i] = i;
    }
    processArray(array, size);
    modifyArray(array);
    for(i = 0; i < size; i++) {
        sum += array[i];
    }
    free(array);
    printf("Sum: %d\n", sum);
    return 0;
}
```

- ① Sum: 40
- ② Sum: 48
- ③ Sum: 50
- ④ Sum: 52

22. 다음은 정점 간의 거리를 표시한 방향 그래프이다. 경로의 길이와 상관없이 이동 가능한 모든 정점 사이의 최단 경로 합은?



- ① 57
② 58
③ 59
④ 60

23. data가 이중 연결 리스트(doubly linked list)에 다음과 같이 구성되어 있을 때, 데이터 A를 삭제한 후 변경된 내용으로 옳은 것은?

memory address	data	left_link	right_link
1000	A	1020	1030
1010	B	1030	NULL
1020	C	NULL	1000
1030	D	1000	1010

- ① B의 right_link → 1030, C의 left_link → 1020
② B의 right_link → 1030, D의 left_link → 1020
③ C의 right_link → 1020, D의 left_link → 1030
④ C의 right_link → 1030, D의 left_link → 1020

24. 다음 키값을 순서대로 삽입하여 B-트리를 구성할 때, 노드 오버플로(overflow)로 인한 노드 분할 횟수는? (단, 차수는 3이다)

1, 5, 7, 11, 13, 15, 17, 19

- ① 1
② 2
③ 3
④ 4

25. 다음 조건에서 데이터가 순서대로 해시 테이블에 입력된다고 할 때, 원래 계산된 버킷(bucket)에 저장되지 않는 데이터의 개수는? (단, 충돌(collision)로 인한 오버플로 발생 시, 선형 조사법(linear probing)으로 처리한다)

- 버킷 크기: 13
○ 슬롯(slot) 크기: 2
○ 해시 함수: $h(k) = k \bmod 13$

21, 33, 8, 19, 76, 58, 27, 83, 30, 44, 7, 73, 2, 66, 85

- ① 1
② 2
③ 3
④ 4