2과목	자료구조 (36 ~ 60)
출제위원 : 방송대 정광식	
출제범위 : 교재 전체 (해당 멀티미디어강의 포함)	

36. 다음 트리에 대한 설명으로 **틀린 것**은 무엇인가? (4점)
① 루트 노드 : 트리에서 부모를 갖지 않은 노드
② 진입 차수 : 트리에 있는 어떤 노드에 대해 그 노드에서 나가는 선의 개수
③ 내부 노드 : 루트도 잎노드도 아닌 노드
④ 형제(sibling) 노드 : 같은 부모를 갖는 노드들

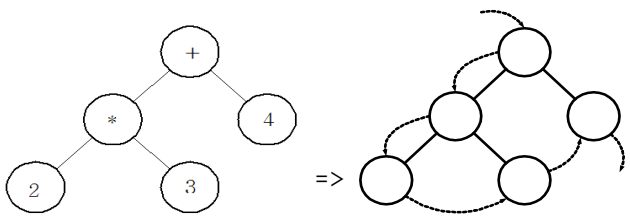
37. 다음은 연결 리스트로 구현한 이진트리의 **중위 순회**를 나타낸 것이다. 빈 칸 **[가], [나], [다]**에 들어갈 가장 적절한 코드는 무엇인가? (2점)

```
   //트리의 노드 정의
struct node {
      struct node *left;
      struct node *right;
      int info;
}
struct node *nodeptr;

//중위 순회
void inorder(struct node *tree_ptr) {
  if (tree_ptr) {
     [가]   ;
     [나]   ;
     [다]   ;
  }
}
```

① [가] :inorder(tree_ptr->left)
[나] :printf("%d", tree_ptr->info)
[다] :inorder(tree_ptr->right)
② [가] :printf("%d", tree_ptr->info)
[나] :inorder(tree_ptr->left)
[다] :inorder(tree_ptr->right)
③ [가] :inorder(tree_ptr->left)
[나] :printf("%d", tree_ptr->info)
[다] :inorder(tree_ptr->left)
④ [가] :inorder(tree_ptr->left)
[나] :inorder(tree_ptr->right)
[다] :printf("%d", tree_ptr->info)

38. 스레드 트리에 대한 설명으로 **틀린 것**은 무엇인가? (4점)
① 잎노드의 사용하지 않는 포인터 부분을 활용할 수도 있다.
② 스레드는 오른쪽 스레드와 왼쪽 스레드 두 가지가 있다.
③ 기존의 트리 노드에 스레드를 위한 포인터를 추가하여 구성할 수도 있다.
④ 오른쪽 스레드는 정해진 순회 순서에 따른 그 노드의 선행 노드를 가리킨다.

39. 다음 이진 트리에 대한 어떤 방문 순서를 스레드로 나타낸 것인가? (2점)



① 깊이 순회 ② 후위 순회
③ 전위 순회 ④ 중위 순회

40. 다음 설명 중에서 **틀린 것**은 무엇인가? (4점)
① 큐 : 먼저 들어간 데이터가 먼저 삭제되는 자료구조
② 우선순위 큐 : 대기 리스트에서 항상 우선순위가 높은 것을 먼저 처리하는 구조
③ 우선순위 큐의 작동 방식 : 삭제 명령이 실행되면 저장된 데이터 중에서 가장 작은 값(가장 큰 값)이 삭제되고, 나머지 데이터들은 특정 순서로 재배열되어 저장되어야 함
④ 최소힙 : 루트가 전체 노드 중에서 최소값인 힙

41. 다음은 힙의 구조체를 정의 한 것과 힙에 노드를 삽입하기 위한 연산을 나타낸 것이다. 빈 칸 **[가]**에 들어갈 가장 알맞은 코드는 무언인가? (2점)

```
//힙 구조체 정의
typedef struct {
    int heap[MAX_Data];
    int heap_size;
} HeapType ;

//힙에 노드를 삽입
void insertHeap(HeapType *h, int item) {
    int i;
    i = ++(h->heap_size);

    while((i != 1) && [가] )      {
        h->heap[i] = h->heap[i/2];
        i /= 2;
    }
    h->heap[i] = item;  }
```

① (item >= h->heap[i/2]) ② (item > h->heap[i/3])
③ (item < h->heap[i/2]) ④ (item < h->heap[i/3])

42. 차례로 정렬된 데이터 리스트 k를 완전한 순서를 유지하는 하나의 리스트로 만드는 과정은 무엇인가? (4점)
① 합병 정렬 ② 선택트리
③ 승자트리 ④ 패자트리

43. 노드 v_i의 왼쪽 서브트리 높이와 v_i의 오른쪽 서브트리 높이가 최대 1만큼 차이가 난다는 조건을 만족하는 트리는 무엇인가? (3점)
① 이진 탐색 트리 ② AVL 트리
③ 병렬 트리 ④ 레드 블랙 트리

44. 모든 키값이 잎에 있고 그 키값에 대응하는 실제 데이터에 대한 주소를 잎 노드만이 가지고 있어서 인덱스된 순차 파일을 구성하는데 사용하는 트리는 무엇인가? (4점)
① B 트리 ② B^* 트리
③ splay 트리 ④ B^+ 트리

45. 다음은 2-3 트리의 노드 구조와 탐색 연산을 나타낸 것이다. 코드에 대한 설명으로 **틀린 것**은? (단, lchild, mchild, rchild는 왼쪽 자식, 중간 자식, 오른쪽 자식을 가리키는 포인터이고, compare(x,t)함수는 검색키 x와 노드 t의 키값을 비교하는 함수이다.) (2점)

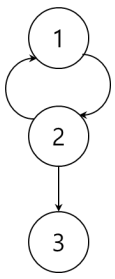
```
// 2-3트리의 노드 구조
typedef struct two_three *two_three_ptr
 struct two_three {
        int lkey, rkey
        two_three_ptr lchild, mchild, rchild
};

// 탐색연산
two_three_ptr search23(two_three_ptr t, int x) {
    while(t)
        switch(compare(x, t)) {
            case 1 : t = t->lchild ;
                     break ;
            case 2 : t = t->mchild ;
                     break ;
            case 3 : t = t->rchild ;
                     break ;
            case 4 : return (t) ;
        }
    return(NULL) ;    }
```

① 2-3 트리 노드의 데이터 구조를 보면 키값을 두 개까지 가질 수 있다.
② compare(x, t)는 x가 비교하는 트리 노드의 키값 중 어느 것과 같은 경우 4를 반환한다.
③ compare(x, t)는 x가 비교하는 트리 노드의 왼쪽 키값보다 크고 오른쪽 키값보다 작은 경우 2를 반환한다.
④ compare(x, t)는 x가 비교하는 트리 노드의 오른쪽 키값보다 큰 경우 1을 반환한다.

46. 다음 그래프에 대한 설명으로 **틀린 것**은 무엇인가? (3점)
① 인접 : 두 정점이 간선으로 연결되었을 때 두 정점은 인접함
② 인접 행렬 : 그래프의 표현방법의 하나로 정점 사이의 인접성을 행렬로 나타낸 것
③ 그래프 : 정점 집합 V와 간선 집합 E에 대하여 그래프는 $G=(V,E)$
④ 간선 : 시작점과 끝점이 같은 경로

47. 그래프에서 특정 정점에서 시작하여 모든 형제를 방문한 후 자손을 방문하는 탐색 방법은 무엇인가? (3점)
① 너비 우선 탐색 ② 깊이 우선 탐색
③ 솔린 탐색 방법 ④ 크루스컬 탐색 방법

48. 다음 [그래프 G] 를 인접 행렬로 표현했을 때, 가장 옳은 것은 무엇인가? (3점)



[그래프 G]

①
	1	2	3
1	0	1	0
2	0	0	1
3	0	0	1

②
	1	2	3
1	0	0	0
2	1	0	1
3	0	1	0

③
	1	2	3
1	0	0	0
2	1	0	1
3	1	0	0

④
	1	2	3
1	0	1	0
2	1	0	1
3	0	0	0

49. 다음 중 그래프에 대한 설명으로 **옳지 않은 것**은? (3점)
① 간선의 시작점과 끝점이 같은 정점의 길이가 1인 경로를 루프 (loop)라고 한다.
② 트리는 그래프의 일종으로, 그래프에서 사이클이 없는 특수한 경우를 말한다.
③ 방향 그래프에서 진입 차수는 주어진 정점에서 시작하는 간선의 개수를 말한다.
④ 무방향 그래프의 차수는 그 정점이 연결된 간선들의 개수이다.

50. 다음 중 스택의 응용에 대한 설명이 **아닌 것**은 무엇인가? (2점)
① 중앙처리 장치 할당을 위한 RR 기법
② 서브루틴의 수행이 끝난 후에 되돌아갈 함수 주소 저장
③ 프로그램에서 사용되는 변수들의 생명주기 관리
④ 연산자들 간의 우선순위에 의해 계산 순서가 결정되는 수식 계산

51. 자료의 복잡한 논리적 성격을 정의하는 형식으로 자료 값의 집합과 연산 집합에 대한 명세의 집합을 무엇이라고 하는가? (3점)
① 추상화 집합 ② 알고리즘
③ 자료형 ④ 추상 자료형

52. 배열에 대한 설명으로 **틀린 것**은 무엇인가? (3점)
① 인덱스와 원소값(<index, value>)의 쌍으로 구성된 집합이다.
② 원소들이 모두 같은 자료형이다.
③ 구성 원소들의 논리적 관계와 원소의 저장 위치는 무관한다.
④ 메모리의 주소값과 추상화된 인덱스값이 관련되어 있다.

53. 아래는 배열값의 저장을 나타낸 것이다. [가]에 들어갈 가장 알맞은 코드는 무엇인가? (2점)

```
1. #define array_size 5
2. void store(int *a, int i, int e) {        // i=3, e=35
3.         if( [가] )
4.                 a[i] = e;
5.         else printf("Error\n");
6. }
```

① i >= 0 && i < array_size
② i >= 0 || i < array_size
③ i <= 0 && i < array_size
④ i <= 0 || i < array_size

54. 파이프의 입구와 출구 부분을 연결시킨 형태의 큐로 기억장소의 낭비를 줄이기 위한 자료구조는 무엇인가? (3점)
① 연결 큐 ② 이중 큐
③ 원형 큐 ④ 데 큐

55. 다음 [가]와 [나]에 알맞은 것은 무엇인가?(2점)

```
void Add_q(int *rear, element item)
{     if  ( [가] )
                {
                   printf("Queue is full !!");
                   return;
                }
       [나]
       return;
  }
```

[가] [나]
① *rear == QUEUE_SIZE-1 queue[++(*front)] = item;
② *rear == QUEUE_SIZE-1 queue[++(*rear)] = item;
③ *front == QUEUE_SIZE-1 queue[++(*front)] = item;
④ *front == QUEUE_SIZE-1 queue[++(*rear)] = item;

56. 다음 중 연결리스트에 대한 설명으로 **틀린 것**은 무엇인가? (3점)
① 데이터가 '논리적인 순서', 혹은 리스트에 나타나는 원소들 간의 '의미적인 순서'를 유지한다.
② 원소의 순서가 메모리 공간에서의 물리적 순서를 의미한다.
③ 원소들의 물리적인 저장 순서나 위치와는 무관하게 원소들 간의 논리적인 순서만 유지해주면 된다.
④ 배열을 이용하여 구현할 수 있다.

57. 다음 10과 3.14를 출력하기 위해 [가]와 [나]에 알맞은 것은 무엇인가? (2점)

```
int a, *p_a;
float b, *p_b;
p_a = [가]
p_b = (float *)malloc(sizeof(float));
*p_a = 10;
*p_b = 3.14;
printf("a is %d,  b is %f\n", [나] );
free(p_a);
free(p_b);
```

[가] [나]
① (float *)malloc(sizeof(float)); *p_a, *p_b

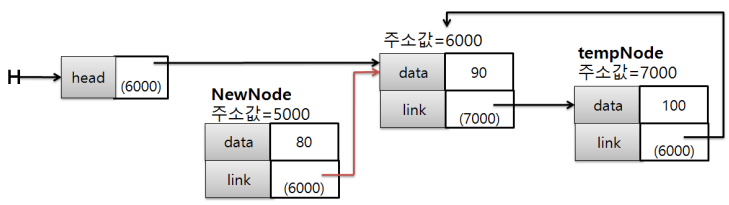
② (int *)malloc(sizeof(int)); &p_a, &p_b
③ (float *)malloc(sizeof(int)); *p_a, *p_b
④ (int *)malloc(sizeof(int)); *p_a, *p_b

※ (58~59) 다음 프로그램에 대한 물음에 답하시오.

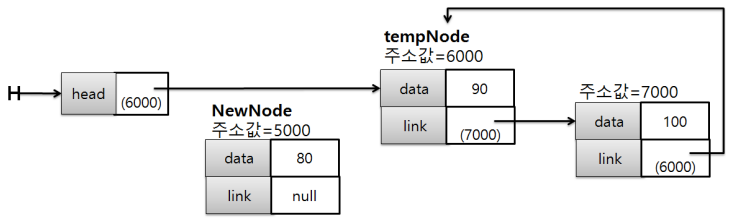
```
① void addFirstNode(linkedList_h* H, int x) {
  //원형 리스트 첫 번째 노드 삽입 연산, x값은 80이라고 가정함
②       listNode* tempNode;
③       listNode* NewNode;

④       NewNode = (listNode*)malloc(sizeof(listNode));
⑤       NewNode → data = x;
⑥       NewNode → link = NULL;

⑦       if (H → head == NULL) { // 현재 리스트가 공백인 경우
⑧             H → head = NewNode;
⑨             NewNode → link = NewNode;
⑩             return;
⑪       }

⑫       tempNode = H → head;
⑬       while(tempNode → link != H → head) tempNode =
  tempNode → link;
⑭       NewNode → link = tempNode → link;
⑮       tempNode → link = NewNode;
⑯       H → head = NewNode;
⑰ }
```

58. 14번 라인을 수행한 결과로 옳은 것은 무엇인가? (2점)

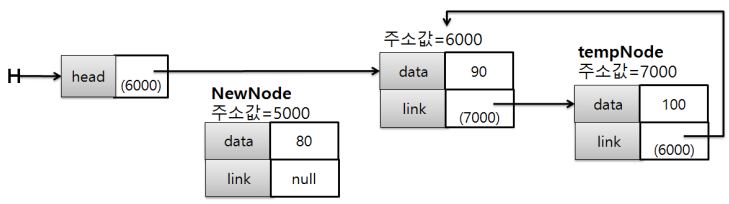
①



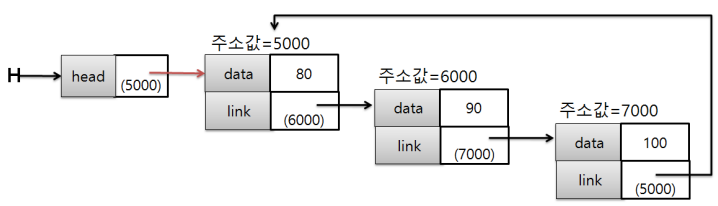
②



③

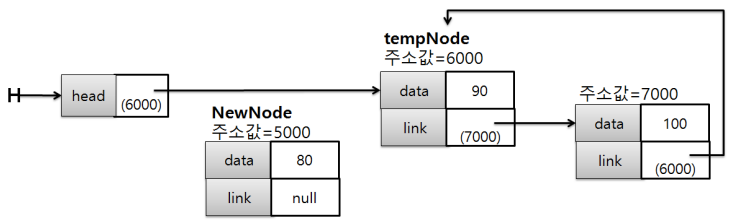


④

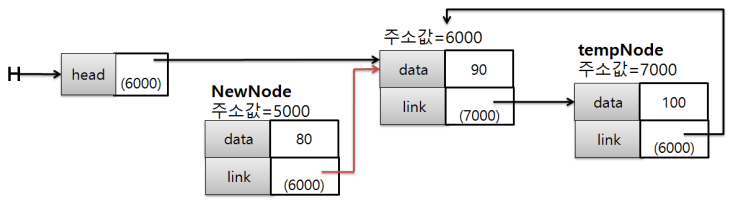


59. 15번 라인을 실행한 결과로 옳은 것은 무엇인가? (2점)

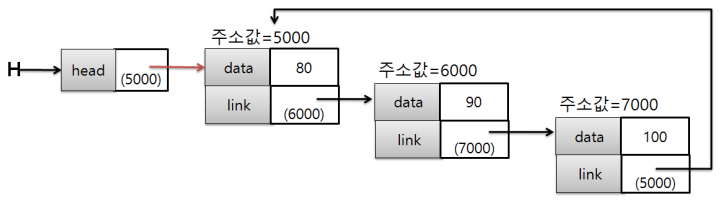
①



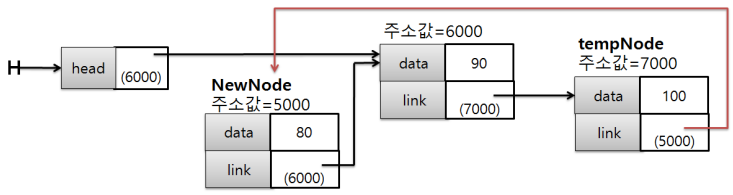
②



③



④



60. 다음은 스택의 연산들이다. ④를 수행한 결과는 무엇인가? (3점)

① CreateS(3);
② Push(stack, 'S');
③ Push(stack, 'T');
④ Pop(stack);
⑤ Push(stack, 'R');
⑥ Push(stack, 'P');
⑦ Push(stack, 'Q');
⑧ Pop(stack);