

자료구조및알고리즘 과제 1

반	1	학과	전자공학과
학번	202021025	이름	안준영

1. 연결 리스트를 구현하는 코드에서 다음의 4개의 함수를 구현하고 구현된 코드를 한글파일에도 추가를 하시오.

<code>void rear_insert_node(ListLinkType *s, ListNode *new_node)</code>
<code>void front_remove_node(ListLinkType *s)</code>
<code>void search(ListLinkType *s, int x)</code>
<code>void concat(ListLinkType *s1, ListLinkType *s2)</code>

- * 구현 함수 안에서 필요한 지역 변수, 포인터 선언하여 사용가능
- * 4개 구현 함수의 함수 원형 수정 금지.
- * 추가적으로 전역 변수, 전역 포인터, 구조체를 선언하여 사용하는 것은 금지
- * 함수 main의 수정 금지.
- * 구현한 코드에 대하여 주석으로 설명.
- * 구현한 코드를 한글 파일에도 추가.

// 생성된 새 노드를 리스트의 맨 뒤에 추가하기.

// 코드를 작성하고 작성한 코드에 대해서 주석으로 설명.

```
void rear_insert_node(ListLinkType* s, ListNode* new_node)
{
    ListNode* x = s->phead; // ListNode 형식의 x를 선언 : 매개변수로 들어온
    리스트 안을 움직일 변수(노드) x

    if(s->phead == NULL) // s->phead가 비어 있는 경우. : new_node가 리스트
    의 처음에 추가되는 경우
        s->phead = new_node; //s의 phead가 new_node의 주소를 가지게 함. : 리스트
    트의 첫 노드

    else // s->phead가 비어있지 않은 경우. : 노드가 추가되는 위치가 리스트의
    처음이 아닌 경우
    {
        while (x->link != NULL) // 어떤 시점의 x가 다음 주소를 가리키지 않
    을 때 까지
        {
            x = x->link; // 다음 노드로 이동
        } // 리스트 끝으로 이동

        x->link = new_node; //x가 리스트 끝에 왔으면 그 때의 x의 link를
    new_node에 이어줌으로써 새로운 노드를 추가.
    }
}
```

// 리스트의 맨 앞의 노드 삭제

// 리스트가 빈 경우(노드가 없는 경우)에는 "삭제오류"라는 메시지만 출력하고 함수가 끝나게 구현(프로그램 종료가 아님).

// 리스트에 노드가 있는 경우에는 맨 앞에 있는 노드 삭제.

// 코드를 작성하고 작성한 코드에 대해서 주석으로 설명.

```
void front_remove_node(ListLinkType* s)
{
    if (s->phead == NULL) //s에 노드가 없어서 오류
    {
        printf("\n삭제오류\n");
    }
}
```

```
else // 삭제할 수 있는 노드가 있는 경우
{
    s->phead = s->phead->link; // 현재 헤드포인터가 가리키는 노드가
    가리키는 노드를 헤드포인터가 가리키게 함으로써 처음 노드(원래 헤드포인터가 가리키던
    노드)를 제거하는 것과 같은 효과.
}
}
```

// 리스트에 있는 데이터를 탐색하여 데이터를 찾은 경우에는 "탐색성공: (찾은 데이터)"이라는 메시지를 출력.

// 같은 값이 리스트에 2개 이상 존재할 수 있으며 이런 경우에는 "탐색성공: (찾은 데이터)"이라는 메시지를 여러 번 출력.

// 데이터를 찾지 못한 경우에는 "탐색실패"라는 메시지를 출력.

// 코드를 작성하고 작성한 코드에 대해서 주석으로 설명.

```
void search(ListLinkType* s, int x)
{
    ListNode* y = s->phead; // ListNode 형식의 x를 선언 : 매개변수로 들어온
리스트 안을 움직일 변수(노드) x
    int i = 0; //탐색성공 or 탐색실패 결정 지을 변수 i
    while (y != NULL) // 매개변수로 들어온 리스트의 노드 중 비어있는 것이 없을
    때 까지
    {
        if (y->data == x) // y가 리스트를 이동하다가 data가 매개변수로 들어
온 x와 같은 노드를 만난 경우
        {
            printf("탐색성공: %d\n", x); // 찾는 데이터 x 출력
            i+=1; // 탐색에 성공하면 i에 1을 더해줌.
        }
        y = y->link; // 다음 노드로 이동
    } // 탐색 과정

    if (i == 0) // i가 0이다 => 위의 while 반복문에서, y가 리스트를 돌아다니면서
x를 만나지 못한 경우. if 조건문에 들어가지 못하여 i는 초기값 0 그대로 있으므로, 탐색
실패
    {
        printf("탐색실패");
    }
}
```

```

// 2개의 리스트 합병: 리스트 S1의 끝에 리스트 S2를 붙여서 합병.
// 두 리스트가 모두 하나 이상의 노드를 가지고 있다고 가정하고 구현.
// 코드를 작성하고 작성한 코드에 대해서 주석으로 설명.
void concat(ListLinkType* s1, ListLinkType* s2)
{
    ListNode* x = s1->phead; // ListNode 형식의 x를 선언 : 매개변수로 들어온
리스트 안을 움직일 변수(노드) x
    // s1, s2 둘 다 하나 이상의 노드가 있으므로 if/else문으로 경우를 나누지 않아도 됨.
    while (x->link != NULL) // 어떤 시점의 x가 다음 주소를 가리키지 않을 때 까지
    {
        x = x->link; // 다음 노드의 링크로 이동
    } // 리스트 끝으로 이동
    x->link = s2->phead; // x의 link에 s2의 헤드포인터를 넣어줌 => s1의 끝 노드의 역할을 하고 있는 x가 s2의 헤드 포인터가 됨으로써 s1과 s2를 이어줌.
}

```

※ 주의사항

1. 과제는 반드시 본인이 작성하여 제출.
2. 타인의 과제를 카피하면 두 과제 모두 미제출로 처리.
3. 제출 기간을 넘기면 감점을 하며 제출 기한에서 1주일이 지난 후에는 과제를 받지 않음.