**README FILE**

**1. void remove\_list**

Remove\_list는 헤드부터 노드를 하나씩 지우고 마지막엔 리스트를 free시켜 전체 리스트를 삭제하는 함수이다. 어떤 값이 지워졌는지 매 노드가 지워질 때마다 그 값을 출력하도록 하였고, 마지막에 총 몇 개의 노드를 삭제했는지 출력하도록 만들었다.

**2. void remove\_node**

Remove\_node는 리스트에서 삭제하고자 하는 값을 삭제하는 함수이다. 삭제하고자 하는 값이 헤드일 때 노드의 숫자가 한 개인지 여러 개인지, 헤드가 아니면 삭제하고자 하는 값이 테일인지 아닌 지로 케이스를 나눠서 구현하였다. 모든 케이스마다 함수가 실행되면 i = 1의 값을 갖도록 하였고, 마지막에 i의 값이 0이면 삭제하고자 하는 값이 리스트의 없다는 문구를 출력하도록 만들었다.

**3. void delete\_range**

Delete\_range는 id\_1부터 id\_2까지있는 모든 노드들을 삭제하는 함수이다. 우선 id\_1과 id\_2의 값을 갖는 노드를 찾도록 구현하였고, 못 찾았다면 에러 문구를 출력하도록 만들었다. 총 4가지 케이스로 분류하여 코드를 작성하였고 만약 id\_1의 순서가 id\_2보다 뒤에 있다면 마찬가지로 에러 문구를 출력하도록 설계하였다.

**4. void push\_Stack**

Push\_Stack은 Stack manner에 의해 스택의 테일에 새로운 노드를 집어넣고 그 노드를 스택의 테일로 할당하는 함수이다. 매뉴얼의 에러 케이스에 맞는 에러 문구를 출력하도록 작성하였다. 스택이 없을 때 push하는 코드는 insert\_node처럼 새로운 리스트를 만들어야 하기 때문에 작성하였다. 이 코드가 없으면 create\_list로 node가 1개인 stack을 만들 때 core dump가 발생한다.

**5. void pop\_Stack**

Pop\_Stack은 스택의 테일에서부터 원하는 개수만큼 하나씩 노드를 삭제하는 함수이다. 지우고자 하는 개수가 노드의 숫자보다 작을 때, 같을 때로 케이스를 나눠서 코드를 작성하였고, 같을 때는 스택을 free시키는 코드를 작성하였다. 또한 매뉴얼에 주어진 에러 케이스에 해당하는 입력이 들어오면 에러 문구를 출력하도록 하였다.

**6. void search\_node**

Search\_node는 찾고자 하는 값을 갖는 노드의 order를 출력해주는 함수이다. while문을 활용하여 헤드 노드부터 하나씩 옮겨가며 원하는 값을 찾고, 찾지 못한다면 에러 문구를 출력하도록 구현하였다.

**7. void swap\_nodes**

Swap\_node는 node1과 node2를 값으로 갖는 2개의 노드의 위치를 바꾸는 함수이다. 이 함수는 총 15개의 케이스로 나눠서 코드를 작성하였다. 마지막으로 앞과 마찬가지로 매뉴얼에 맞게 에러 케이스에 해당하는 문구를 출력하도록 하였다.

**8. void forward\_by\_one**

Forward\_by\_one은 리스트의 헤드를 테일로 옮기고 나머지 노드들은 하나씩 왼쪽으로 이동시키는 함수이다. 함수가 실행될 때마다 헤드와 테일이 다시 할당되도록 구현하였고 에러 케이스에 맞는 문구를 출력하도록 설계하였다.

**9. void backward\_by\_one**

Backward\_by\_one은 리스트의 테일을 헤드로 옮기고 나머지 노드들은 하나씩 오른쪽으로 이동시키는 함수이다. 함수가 실행될 때마다 마찬가지로 헤드와 테일이 재할당되고 에러 케이스가 발생하면 에러 문구 또한 출력하도록 하였다.

**10. void reverse\_range**

Reverse\_range는 order1부터 order2까지에 포함된 노드들의 순서를 뒤집는 함수이다. Order1과 order2사이에 있는 노드들의 개수가 홀수인지 짝수인지로 케이스를 나눠서 코드를 작성하였다. 또한 이 경우도 케이스가 매우 많기 때문에 코드의 양을 줄이고자 swap\_node함수를 이용하였다.

**11. void powerofN\_reverse**

powerofN\_reverse함수는 입력 값 n의 power만큼 묶어서 노드들의 순서를 뒤집는 함수이다. 마찬가지로 reverse\_range함수를 이용하였고 pow(n,i) 함수를 이용하여 코드를 작성하였다. 함수의 실행은 두가지 단계로 나눴는데 우선 for문은 pow(n,i)의 숫자에 맞게 노드들을 묶어서 reverse시키고 그 다음 if문은 pow(n,i) 값보다 reverse하지 않은 남아있는 노드들의 수가 더 적어 묶을 수 없을 때 나머지들을 어쨌든 reverse시키는 코드이다.