**2024 Spring OOP Assignment Report**

과제 번호 : Assignment #3 - Prob1

학번 : 20230563

이름 : 김홍근

Povis ID : hongsimi7

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

1. **프로그램 개요**
   * 본 프로그램은 Linked list를 이용해 element를 추가하고, 삭제하고, 접근하고, 값이 list에 존재하는지 확인하는 기능을 하는 프로그램이다.
   * 각 element는 오름차순으로 정렬되어 OrderedList에 추가된다.
   * Structure을 사용하여 값을 저장하고 Linked list를 구현한다.
   * Pointer를 사용하여, 본 프로그램의 주요 기능을 구현한다.
2. **프로그램의 구조 및 알고리즘**

* add 함수는 overloading을 이용하여, 함수를 두 가지로 정의한다. 단일 값을 받을 때와 배열 값을 받을 때로 나뉜다
* 단일 값을 받는 add 함수는 리스트에 단일 값을 추가하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered와 정수형의 v를 가진다. ordered은 Linked list의 head를 가리키고, v는 추가할 데이터이다. 동적할당을 이용하여, 새로 추가할 노드를 newNode라고 정의한다. newNode는 Node형이고, data에 v를 저장한다. Node형의 findNode를 선언하여 첫번째 node를 가리키게 하고, Node형의 선언된 preNode는 nullptr를 가리키게 한다. 이 때, preNode는 findNode의 이전 노드를 의미한다. while문을 사용하여, preNode와 findNode를 한 노드씩 앞으로 옮기며, v가 들어갈 알맞은 자리를 찾는다. If-else문을 이용하여, newNode가 head가 되어야 할 때와 아닐 때를 나눈다. Head가 되어야 할 때에는 newNode가 ordered와 원래 head였던 노드 사이에 연결된다. 아닐 때에는 preNode와 findNode사이에 연결된다. size함수 호출하여 parameter로 ordered를 보내어, ordered의 m\_size를 변경한다.
* 배열 값을 받는 add 함수는 리스트에 배열 값을 추가하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered와 상수형정수 배열 포인터의 arr와 정수형 size를 가진다. ordered은 Linked list의 head를 가리키고, arr는 추가할 데이터 배열, size는 배열의 크기이다. for문을 사용하고 단일 값을 받는 add 함수를 호출하여, 배열의 요소를 하나씩을 parameter로 보내어 list에 추가한다.
* remove함수는 리스트에서 인덱스의 값을 삭제하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered와 정수형의 index를 가진다. ordered은 Linked list의 head를 가리키고, index는 삭제할 데이터의 배열 인덱스이다. 동적할당을 이용하여, 새로 추가할 노드를 newNode라고 정의한다. newNode는 Node형이고, data에 v를 저장한다. Node형의 findNode를 선언하여 첫번째 node를 가리키게 하고, Node형의 선언된 preNode는 nullptr를 가리키게 한다. 이 때, preNode는 findNode의 이전 노드를 의미한다. For 문을 사용하여, findNode는 삭제할 노드를 가리키고, preNode는 그 노드의 이전 노드를 가리킨다. If-else문을 이용하여, preNode가 가리키는 노드가 있는지 없는지 구별한다. 만약 있다면, preNode는 findNode의 다음 노드를 가리키고, 아니면 ordered이 findNode의 다음 노드를 가리키게 한다. 그리고 findNode를 동적 할당 해제해주고, size함수 호출하여 parameter로 ordered를 보내어, ordered의 m\_size를 변경한다.
* size 함수는 리스트의 크기를 반환하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered를 가진다. ordered은 Linked list의 head를 가리킨다. 노드를 개수를 셀 정수형 count를 선언하고, 현재 카운트한 노드를 가리킬 countNode를 선언하고, head 노드를 가르키게 한다. while문을 사용하여, countNode가 nullptr를 가르킬 때까지 한 노드씩 이동하며, count에 +1을 한다. while문이 끝나면, count를 반환한다.
* contains 함수는 리스트에 특정 값이 있는지 확인하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered와 정수형의 v를 가진다. ordered은 Linked list의 head를 가리키고, v는 찾고 싶은 값이다. Node형의 findNode를 선언하여 첫번째 node를 가리키게 한다. while문을 사용하여 findNode가 nullptr을 가리킬 때까지 반복을 하는데, 이때 if-else문 사용하여 만약 findNode가 가리키는 노드의 데이터가 v와 같을 때, true를 반환하고, 아니면 다음 노드로 넘어간다. 반복문이 끝났다면 찾는 값이 없으므로 false를 반환한다.
* getValue는 리스트에서 인덱스의 값을 반환하는 함수이다. parameter로 OrderedList의 포인터 형식의 ordered와 정수형의 idx를 가진다. ordered은 Linked list의 head를 가리키고, idx는 보고 싶은 list의 요소 번호이다. if문을 사용하여, 만약 idx가 list 크기보다 클 경우, 가장 작은 정수를 출력한다. Node형의 getValueNode를 선언하여 첫번째 node를 가리키게 한다. for문을 사용하여, idx만큼 getValueNode를 다음 node로 이동시킨다. 반복문이 끝났다면, getValueNode가 가리키는 노드의 데이터를 반환한다.

1. **토론 및 개선**
   * + 구조체를 생성하는 방법 및 구조체를 유용하게 사용하는 방법에 대해 알 수 있었다. 구조체에 포인터를 넣어 다음 구조체를 가리키는 것으로 Linked List를 만들 수 있다는 것을 알 수 있었다.
     + Linked List의 정의와 형태 및 종류에 대해 알 수 있었으며, 삽입 및 삭제, 확인 등등의 기능을 구현할 수 있었다.
     + 함수를 만들어 직관적인 코딩을 가능하게 했다.
     + 함수를 오버로딩하여 같은 이름의 함수가 비슷한 기능을 할 수 있도록 했다.
     + 동적 할당을 통해 불필요한 메모리 소모를 막으며, Linked List를 구현하여 프로그램을 제작할 수 있었다.
     + Pointer 관계에 대해 자세히 생각한 나머지, 프로그램 속도 및 메모리 소모를 크게 신경 쓰지 못했다. 조금 더 고민을 해본다면, 더 간결하고 빠른 속도의 코딩이 가능할 것이라 생각했다.
2. **참고 문헌**
   * + 해당사항 없음.