**2024 Spring OOP Assignment Report**

과제 번호 : 3-2

학번 : 20230499

이름 : 김재환

Povis ID : 20230499

**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**

* OOP Assignment #3 – 문제 2번 (배점 30점)

주어진 설명에 따라 정렬된 형태의 정수 set을 구현하고, 그에 맞는 매개변수와 반환 타입을 갖는 함수들을 구현하는 것이 목적이다. 구현하고자 하는 함수들은 다음과 같다. Add, remove, size, contains, getValue. 각 함수에 대한 설명은 2. 프로그램의 구조 및 알고리즘에서 설명한다. 1번 문제인 Orderedlist와 매우 유사하나, 삽입시 중복이 있는지 확인하는 알고리즘만 추가하면 된다.

1. **프로그램의 구조 및 알고리즘**

* 프로젝트를 빌드할 때 중심이 되는 main.cpp에 모두 구현하지 않고, 사용자 정의 헤더파일 prob1.h에서 구조체 및 함수 프로토타입을 선언하였으며, 소스코드 prob1.cpp에서 함수를 구현하였다. 이후 prob1.h를 include하여 사용하였다.
* OrderedSet는 구조체를 통해 singly-linked list 방식으로 구현되며, 구조체 내 변수로 int형 변수인 m\_size와 Node 구조체 포인터 변수 head를 갖는다. Node 구조체는 해당 노드가 갖는 값과 다음 노드를 가르키는 Node 포인터 변수 next를 갖는다. OrderedSet 선언시 head는 nullptr을 가르키며, 이후 add함수가 실행되고 난 후에는 head가 첫 노드를 가르킨다. 코드를 보면 이해하기 쉬울 것이다.
* Add 함수는 매개변수에 따라 다른 동작을 하도록, 다형성을 가진다. 이는 함수 오버로딩을 통해 구현된다. 첫 번째로, 매개변수로 OrderedSet 포인터와 int형 정수 하나를 받을 경우, contains함수를 통해 해당 정수가 이미 set 안에 존재하는지 확인한 후, 없다면 new 키워드를 통해 노드 하나를 할당받은 후 OrderedSet의 head에서부터 앞으로 순회하면서 각 원소가 오름차순이 되도록 적절한 위치에 연결한다. 매개변수로 OrderedSet 포인터와 다른 배열 포인터, 배열의 크기를 나타내는 정수를 입력받을 경우, 다른 배열을 가르키는 포인터를 크기 만큼 순회하면서 OrderedSet에 삽입한다. 이 과정에서 첫 번째에 구현된 add 함수를 사용하였다
* Remove 함수는 주어진 인덱스에 해당하는 노드를 삭제한다. 의 head로부터 index까지 반복하면서 나아가고, 찾은 위치를 새 노드 포인터 del에 저장한다. 이후 del 노드를 제외하고 다른 노드를 재 연결한 이후, delete 키워드를 통해 del 노드를 지운다. new키워드로 메모리 영역을 할당받았으므로 delete 키워드를 사용하지 않으면 메모리 누수가 발생할 수 있다.
* Size 함수는 집합의 원소 개수를 반환한다. OrderedSet의 m\_size 값을 반환한다.
* Contains함수는 매개변수로 받은 변수가 집합 내에 존재하는지 확인한다. OrderedSet의 head로부터 size만큼 순회하면서 찾는 값이 집합 내에 존재하는지 확인한다. 초기 어싸인 요구사항에서 제시된 isDup 함수는 contains 함수와 동일한 함수이다.
* getValue함수는 주어진 index에 해당하는 값을 반환하는 함수이다. 매개변수로 받은 index가 집합의 사이즈보다 크다면 INT\_MIN을 반환한다. 인덱스가 유효하다면 head에서 출발해서 index를 찾은 후 값을 반환한다.

1. **토론 및 개선**
   * + OrderedSet의 최대 크기가 정해져 있지 않으므로, new와 delete를 통해 동적할당으로 구현하였다.
     + 시간 복잡도를 분석하면 다음과 같다. O는 OrderedSet이다. add(o, value)는 head로부터 값이 들어갈 위치를 찾아야 하므로 O(n), add(o, \*arr, arr\_size)는 add(o, value)를 arr\_size만큼 실행하므로, O(n\*arr\_size)이다. Size(o)는 변수 값 하나를 반환하므로 O(1)이다. Remove(o, idx), Contains(o, v)와 getValue(o, idx)는 모두 head에서부터 원하는 위치를 찾아야하므로 시간복잡도는 O(n)이다.
2. **참고 문헌**
   * + Prata, S. (2013). C Primer Plus (6th ed.).