4주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211558 이름: 윤준서

**1. 목적**

객체 지향 프로그래밍은 상속과 더불어, 다형성(Polymorphism)을 제공함으로써 프로그래밍의 효율성과 유연성 및 재사용성을 달성한다. C++에서 제공하는 몇몇 방법들을 통해 객체지향 프로그래밍의 이점을 살펴볼 수 있다. 이번 실험에서는 상속과 더불어 교재에서 언급한 다형성의 예를 살펴봄으로써 효율적인 프로그래밍을 위한 기술을 익힌다.

**2. 알고리즘과 자료구조**

여러 종류의 자료형을 받기 위해 template 자료형을 LinkedList 와 Stack 을 정의할 때 사용한다.

함수명과 변수 자료형에 template <typename T> 와 T 자료형을 삽입한다. 이 때 Node 와 LinkedList 를 클래스를 사용해 정의하고, 이들을 상속받는 Stack 을 정의한다.

--------------------------------------------------------------------------------------------------------------

tmeplate <class T> class Node

public

**T data** 노드의 값을 저장하는 데이터다.

**Node \*link** 다음 노드의 주소를 저장하는 포인터다.

**Node(T element)** element 를 노드와 주소를 저장하는 구조체로 정의한다.

template <class T> class LinkedList

protected

**Node<T> \*first** 연결 리스트의 첫 노드를 참조하고 저장한다.

**int current\_size** 연결 리스트의 크기를 저장한다.

public

**LinkedList()** 연결 리스트의 생성자다.

**int GetSize()** current\_size 를 반환한다.

**void Insert(T element)** 원소를 저장하고, 새 노드를 생성하여 연결 리스트의 첫 번째 노드로 설정한다.

**virtual bool Delete(T element)** 연결 리스트의 마지막 노드를 탐색하여 제거한다. 이 후 current\_size 를 1 감소시킨다.

**void Print()** 연결 리스트의 모든 data 를 출력한다.

--------------------------------------------------------------------------------------------------------------

class Stack : public LinkedList <T>

public

**bool Delete(T &element)** 스택의 첫 노드(가장 최근에 추가된 노드)를 제거한다. 이후 두 번째 노드를 첫 번째 노드로 설정한다. 이후 current\_size 를 1 감소시킨다.