4주차 예비보고서

전공: 수학과,컴퓨터공학과 학년: 3학년 학번: 20181256 이름: 김도현

1.1

이 문제는 queue(first in first out) 형태의 연결 리스트인 LinkedList 인 기반클래스에서 파라미터적 다향성(Parametric Polymorphism)을 통해 데이터를 int 형 뿐만 아니라 다양한 데이터 유형을 받을 수 있도록 해주는 것이다. 또한 LinkedList 클래스를 상속하는 Stack(last in first out) 파생 클래스를 만들어서 Stack 역시 구현 가능해야 한다.

우선 각 리스트의 노드들은 Node 클래스로 구현한다. 여기에는 받아온 데이터를 저장하는 data, 연결 노드 포인터를 가지고 있는 Node \*link가 있다. 이 node를 이용해 LinkedList 클래스를 구현할 수 있다. 멤버변수 Node \*first(노드의 first를 의미함), int current\_size(노드들의 개수) 는 protected로 접근지정자로 하여 후에 파생되는 클래스에서도 이용 가능하게 한다. 그리고 생성자에서는 멤버변수들을 초기화 한다. 멤버함수인 int GetSize() 에서는 current\_size를 return 하여 현재의 노드가 몇개인지 알 수 있다. void Insert(element) 에서는 새로운 노드를 만들어 element 를 data로 저장하고 새 노드를 원래의 first 노드에 연결하고 새 노드를 first 로 바꾼 후 current\_size를 하나 올린다. Bool Delete(&element) 에서는 first 가 존재한다면 current 노드와 previous 노드를 이용하여 while 문을 통해 가장 먼저 들어온 노드를 찾고 이를 삭제 한 후 다시 previous-link를 current->link로 바꾸거나 previous가 없으면 first=first->link 로 한다. 이후 삭제 되는 노드의 data를 element 에 저장하고 current\_size를 하나 줄인다. Print() 에서는 반복문을 통해 연결된 노드들을 처음부터 끝까지 순차적으로 각 노드들의 data 를 출력해 준다.

이후 효율적인 코드 사용을 위해 여러 데이터 유형을 받을 수 있게 템플릿 이라는 자료형(T)을 데이터 변수들에 사용한다. 그리고 클래스 선언부에서 template <class T> 를 적어 템플릿 클래스를 구현한다. (ex. Insert(T element) , string 데이터형 이용 시 헤더 삽입부(#include <string>)

Stack 클래스는 LinkedList 클래스를 상속받아 대부분을 그대로 사용할 수 있으나 Stack의 성질인 last in first out 은 따로 구현 해야 한다. 즉 delete 함수를 다시 구현 하면 된다.(Overriding) 이때 처음 생긴 노드가 삭제되어야 하기 때문에 첫 노드와 연결되 있는 다음 노드를 first 로 한 다음 element에는 첫 노드의 data, 이후 첫 노드를 delete 하고 current\_size를 하나 줄인다. 참고로 Stack을 구현할 때 first나 current\_size 같은 class의 멤버 변수는 this 포인터를 이용한다. delete 함수처럼 재정의 하는 함수는 virtual로 선언하여 정상 작동 할 수 있게 한다. 즉 virtual 선언 시 LinkedList클래스 의 pointer, reference 를 통해 Stack 클래스 접근이 가능하다. (서브타입 다형성)