전공 : 컴퓨터공학과 학년 : 4 학번 : 20212020 이름 : 박민준

1. 실험 시간에 작성한 프로그램의 알고리즘과 자료구조를 요약하여 기술하시오.

4주차 실습에서는 C++을 사용하여 객체 지향 프로그래밍의 상속과 다형성을 다루었는데, 이를 위해 2개의 중요한 자료구조인 LinkedList와 Stack 클래스를 구현하였다. 특히 LinkedList 클래스는 c++ template를 사용하여 다양한 데이터 타입을 저장할 수 있도록 구현하였다. 각 노드는 데이터와 다음 노드를 가리키는 포인터로 이루어져 있기 때문에 이러한 구조를 통해 데이터를 삽입하고, 삭제하고, 탐색하는 등의 연산을 효율적으로 수행할 수 있다.

1) LinkedList 클래스

[자료구조]

LinkedList는 각 노드들이 선형적으로 연결된 구조로, 먼저 Node 구조체가 어떻게 구현되어 있는지를 설명하고자 한다. 각 노드는 데이터 값을 갖는 data 변수와 다음 노드를 가리키는 link Node(포인터)를 멤버 변수로 가진다. 이 때, data 변수의 타입은 template 파라미터인 ‘T’에 의해 결정된다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[사진1. Node class]

이러한 노드를 이용한 LinkedList 클래스는 List의 첫 번째 노드의 주소를 저장하는 first 포인터와, 현재 list의 크기를 나타내는 current\_size 변수를 포함한다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[사진2. LinkedList class]

[알고리즘]

LinkedList의 생성자는 first Node와 current\_size의 값을 0으로 초기화하고, GetSize() 함수는 노드의 전체 개수를 반환하는 함수이다.

이후 데이터의 삽입, 삭제, 출력을 위해 Insert, Delete, Print 함수를 사용하였다.

void Insert(T element) 함수는 새로운 노드를 list의 맨 앞에 삽입한다. 만약 current\_size가 임의로 설정한 노드의 최대 개수 MAX 값과 같다면 list에 더 이상 노드를 삽입할 수 없다는 의미이므로 노드를 삽입하지 않고 반환한다. 그렇지 않다면 새로운 노드의 link는 현재 첫 번째 노드를 가리키고, list의 first 포인터가 새로운 노드를 가리키도록 만들어 노드를 삽입한다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

[사진3. Insert 함수]

virtual bool Delete(T &element) 함수는 list의 마지막 노드를 삭제하고 해당 데이터를 반환한다. 만약 list의 첫 번째를 가리키는 first가 0이라면 해당 LinkedList는 비어 있다는 의미이므로 false를 반환한다. list가 비어 있지 않다면, list의 마지막 노드를 찾기 위해 while문을 이용하여 list를 처음부터 순회하며, 마지막 노드의 link를 0으로 설정한다. 이 때, if문 안의 if문은 노드의 개수가 2개 이상인 경우이고, else문은 노드의 개수가 1개인 경우를 의미한다. 노드의 개수가 1개일 때는 list가 비어 있도록 만들어야 하기 때문에 first 포인터의 값을 0으로 설정하게 된다.

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

[사진4. Delete 함수]

마지막으로 void Print() 함수는 list의 모든 요소를 순회하며 값을 출력해주는 함수이다. 만약 list가 비어 있다면 아무것도 출력하지 않고 함수를 종료한다.

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

[사진5. Print 함수]

이러한 LinkedList 클래스는 template을 사용하여 구현되기 때문에 어떠한 타입의 데이터도 저장할 수 있는 장점을 가지고 있다.

2) Stack 클래스

[자료구조]

Stack은 후입선출(LIFO) 방식으로 작동하는 선형 자료구조인데, 해당 실습에서는 LinkedList를 상속받아 구현하였다. Stack 클래스는 LinkedList의 클래스를 상속받기 때문에 해당 클래스의 노드 구조, 첫 번째 노드를 가리키는 first 포인터, 리스트의 크기를 나타내는 current\_size 등의 변수와 메소드 등 모든 속성을 포함한다. 따라서 Stack은 LinkedList와 동일한 자료구조를 갖는다.

[알고리즘]

Stack에서 요소의 삽입은 Stack의 최상단에 새로운 요소를 추가하는 방식으로 구현된다. 이는 LinkedList의 Insert() 함수를 그대로 사용하여 새로운 노드를 list의 맨 앞에 삽입함으로써 구현할 수 있다. 이러한 방식은 Stack의 push 연산에 해당된다.

Stack에서 요소의 삭제는 Stack의 최상단 요소를 삭제하고 해당 데이터를 반환한다. Stack 클래스의 알고리즘은 LinkedList와 거의 유사하지만, Delete() 함수에서 차이를 보인다. Stack 클래스는 LinkedList 클래스와 다르게 후입선출 방식을 갖기 때문에 노드를 삭제할 때 맨 뒤 요소가 아닌 맨 앞 요소를 삭제한다. 이는 LinkedList의 Delete() 함수를 Overriding하여 구현할 수 있으며 list의 첫 번째 노드를 삭제하고 해당 데이터를 element 변수에 저장한다. 이는 스택의 pop 연산에 해당된다.

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

[사진6. Stack class에서 Delete 함수 Overriding]

2. 참고자료

1) 4주차\_기초 C++ 프로그래밍 #2.pdf

2) 4주차\_CPP-2\_다형성의 이해.pdf