2015 자료구조 실습과제 10

1. 배열을 이용한 리스트 구현

(1) 배열을 이용한 리스트 클래스를 구현하시오.

* 강의시간에 설명한 바와 같이 구현

* 강의자료 (pdf파일) 과 유사하게 처리하면 됨.

* 클래스 이름에 각자의 이니셜을 추가할 것

2. Wish List의 구현

(1) 사고 싶은 물품 항목을 표현하는 리스트를 단순 연결 리스트로 구현하시오.

(2) Item 클래스

- 물품의 이름, 물품의 가격
- 생성자, 소멸자, 화면 출력, 비교 함수 등을 포함

(3) ItemNode 클래스: Item 클래스에서 상속받아서 구현

- 스택과 큐에서의 Node 클래스와 유사함
- AddNext()와 DeleteNext() 함수 구현

(4) CLinkedList 클래스

- 헤드 노드를 이용해서 구현
- 강의자료의 모든 멤버들을 구현할 것
- Merge()와 Reverse()는 구현하지 않아도 됨.
- 첨부의 main()함수와 유사하게 구현된 기능을 테스트 할 것

```
// 파일명: LinkedList.h : 단순 연결 리스트 클래스
#include "Node.h"
class CLinkedList
{
    CNode   m_org;                  // 헤드 노드 (헤드 포인터가 아님)
public:
    CLinkedList(void): m_org(0) { }  // 생성자
    ~CLinkedList(void) { Clear(); }  // 소멸자

    CNode* GetHead()        { return m_org.Next(); }
    bool IsEmpty( )         { return GetHead()==NULL; }

    // 리스트의 마지막 항목을 반환. 단, 리스트가 비어있으면 헤드노드의 주소를 반환
    CNode* GetTail() {
        CNode* tail = &m_org;
        for( ; tail->Next() != NULL ; tail = tail->Next() );
        return tail;
    }

    void Display( ) {
        printf( "[전체 항목 수 = %2d] : ", GetLength());
        for( CNode *p = GetHead() ; p != NULL ; p=p->Next() )
            p->Display();
        printf( "\n");
    }

    // CLinkedList 클래스의 삽입 함수들
    void AddHead( CNode *n ) { ... }          // 맨앞 삽입: 프로그램 4.8
    void AddTail( CNode *n ) { ... }          // 맨뒤 삽입: 프로그램 4.8
    void AddNext(CNode *p, CNode *n ) { ... } // 중간 삽입: 프로그램 4.8

    // CLinkedList 클래스의 삽입 함수들
    void DeleteHead() { ... }                 // 맨앞 삭제: 프로그램 4.9
    void Delete( CNode *n ) { ... }           // n 삭제: 프로그램 4.9
    void Clear() { ... }                      // 모두 삭제: 프로그램 4.9

    // CLinkedList 클래스의 탐색 함수들
    CNode *FindPrev( CNode *n ) { ... }       // 프로그램 4.10
    CNode *FindNode( int val ) { ... }        // 프로그램 4.10

    // 기타 함수들
    void Merge( CLinkedList &list2 ) {... }   // 프로그램 4.12
    void Reverse( ) { ... }                   // 프로그램 4.13
};
```

```
// 파일명: 4장-LinkedList.cpp : 단순 연
#include "LinkedList.h"
void main()
{
    CLinkedList     list;
    list.AddHead( new CNode(10) );
    list.AddTail( new CNode(20) );
    list.AddTail( new CNode(30) );
    list.AddTail( new CNode(40) );
    list.Display();

    CNode *n = list.FindNode( 20 );
    list.AddNext(n, new CNode(50));
    list.Display();

    list.DeleteHead();
    n = list.FindNode( 30 );
    list.Delete( n );
    list.Display();

    list.Reverse( );
    list.Display();

    CLinkedList     list2;
    list2.AddHead( new CNode(80) );
    list2.AddHead( new CNode(70) );
    list2.AddHead( new CNode(60) );
    list2.Display();

    list.Merge( list2 );
    list.Display();
    list2.Display();

    list.Clear();
    list.Display();
}
```

출력 화면

```
[전체 항목 수 =  4] :  [ 10] [ 20] [ 30] [ 40]
[전체 항목 수 =  5] :  [ 10] [ 20] [ 50] [ 30] [ 40]
[전체 항목 수 =  3] :  [ 20] [ 50] [ 40]
[전체 항목 수 =  3] :  [ 40] [ 50] [ 20]
[전체 항목 수 =  3] :  [ 60] [ 70] [ 80]
[전체 항목 수 =  6] :  [ 40] [ 50] [ 20] [ 60] [ 70] [ 80]
[전체 항목 수 =  0] :
[전체 항목 수 =  0] :
```