Intermediate Programming

Day 37

Outline

- Exercise 13-2

Exercise 13-2 (part 3)

- Complete the implementation of MyList< T >::iterator

```
                    MyList.h
...
template< typename T >
class MyList
{
...
        class iterator
        {
                MyNode<T> *ptr;
        public:
                iterator( MyNode<T> *initial ) : ptr(initial) { }
                ...
        };
        ...
```

Exercise 13-2 (part 3)

- Complete the implementation of MyList< T >::iterator

MyList.h

```
...
template< typename T >
class MyList
{
...
        class iterator
        {
                MyNode<T> *ptr;
        public:
                iterator( MyNode<T> *initial ) : ptr(initial) {
                ...
        };
        ...
```

MyList.h

```
...
template<typename T>
class MyList
{
...
        class iterator
        {
                MyNode<T> *ptr;
        public:
                iterator( MyNode<T> *initial ) : ptr(initial) { }
                iterator &operator++() { ptr=ptr->next ; return *this; }
                bool operator!=(const iterator &o) const { return ptr!=o.ptr; }
                T &operator*(){ return ptr->data; }
        };
        iterator begin( void ){ return iterator(head); }
        iterator end( void ) { return iterator(nullptr); }
        ...
```

Exercise 13-2 (part 4)

- Implement MyList< T >::const_iterator

MyList.h

```
...
template< typename T >
class MyList
{
...
	class const_iterator
	{
		...
	};
	...
```

MyList.h

```
...
template< typename T >
class MyList
{
...
	class const_iterator
	{
		const MyNode<T> *ptr;
	public:
		const_iterator( const MyNode<T> *initial ) : ptr(initial) { }
		const_iterator& operator++() { ptr=ptr->next ; return *this; }
		bool operator!=( const const_iterator &o ) const { return ptr!=o.ptr; }
		const T &operator*(){ return ptr->data; }
	};
	const_iterator cbegin( void ) const{ return const_iterator(head); }
	const_iterator cend( void ) const { return const_iterator(nullptr); }
	...
```

Exercise 13-2 (part 5)

- Implement the `MyList< T >::MyList( Itr i_begin , Itr i_end )` constructor

MyList.h

```
...
template<typename T>
class MyList
{
...
        template<typename Itr>
        MyList( Itr i_begin , Itr i_end )
        {
        }
        ...
```

MyList.h

```
...
template<typename T>
class MyList
{
...
        template< typename Itr >
        MyList( Itr i_begin , Itr i_end ) : head(nullptr)
        {
                for( Itr i=i_begin ; i!=i_end ; i++ ) insertAtTail( *i );
         }
        ...
```

Final Project

- Website -> Assignments -> Final Project