

실습과제#6 : 문자열 덧셈

실습과제 5를 기반으로 실습과제 6을 진행합니다.

■ GetLength() 메서드 추가

CMyString 클래스의 CMyString.h 와 CMyString.cpp 파일에 다음과 같이 public 접근 제어 지시자를 적용한 GetLength() 메서드를 추가합니다.

CMyString.h	CMyString.cpp
<pre>1 #pragma warning(disable:4996) 2 #pragma once 3 class CMyString 4 { 5 public: 6 CMyString(); 7 ~CMyString(); 8 // 변환생성자 9 explicit CMyString(const char* szPa 10 11 // 복사생성자 12 CMyString(const CMyString& rhs); 13 14 // 이동생성자 15 CMyString(CMyString&& rhs); 16 17 int GetLength() const; 18</pre>	<pre>48 int CMyString::GetLength() const 49 { 50 return m_nLength; 51 }</pre>

■ Append() 메서드 추가

함수 추가

함수 이름(U)

Append

반환 형식(Y)

int

엑세스(A)

public

.cpp 파일(F)

CMyString.cpp

매개 변수(P)

const char* pszParam

주석(M)

기타 옵션:

☐ 인라인(I)

☐ 정적(S)

☐ 시각적 개체(V)

☐ 순수(P)

확인

취소

CMyString.h

26 public:
27 int SetString(const char* pszParam);
28 const char* GetString();
29 void Release();
30 int Append(const char* pszParam);
31 };
32

CMyString.cpp

112 int CMyString::Append(const char* pszParam)
113 {
114 // TODO: 여기에 구현 코드 추가.
115 return 0;
116 }
117

■ Append() 메서드 정의 (this객체 m_pszData 에 pszParam 문자열을 합치는 함수)

```
int CMyString::Append(const char* pszParam)
{
    // 매개변수 유효성 검사
    if (pszParam == NULL)
        return 0;

    int nLenParam = strlen(pszParam);
    if (nLenParam == 0)
        return 0;

    // 세트된 문자열이 없다면 새로 문자열을 할당한 것과 동일하게 처리함
    if (m_pszData == NULL)
    {
        SetString(pszParam);
        return m_nLength;
    }

    // 현재 문자열의 길이 백업
    int nLenCur = m_nLength;

    // 두 문자열을 합쳐서 저장할 수 있는 메모리를 새로 할당함
    char* pszResult = new char[nLenCur + nLenParam + 1];

    // 문자열 조합
    strcpy(pszResult, m_pszData);
    strcpy(pszResult + (sizeof(char) * nLenCur), pszParam);

    // 기존 문자열 삭제 및 멤버 정보 갱신
    Release();

    m_pszData = pszResult;
    m_nLength = nLenCur + nLenParam;

    return m_nLength;
}
```

■ + 연산자함수, += 연산자함수 추가

CMyString 클래스의 CMyString.h 와 CMyString.cpp 파일에 CMyString operator+(const CMyString& rhs), CMyString& operator+=(const CMyString& rhs) 연산자 함수를 추가합니다.

CMyString.h	CMyString.cpp
<pre> class CMyString { public: CMyString(); ~CMyString(); // 변환생성자 explicit CMyString(const char* szParam); // 복사생성자 CMyString(const CMyString& rhs); // 이동생성자 CMyString(CMyString&& rhs); int GetLength() const; CMyString operator+(const CMyString& rhs); CMyString& operator+=(const CMyString& rhs); </pre>	<pre> 53 CMyString CMyString::operator+(const CMyString& rhs) 54 { 55 return CMyString(); 56 } 57 58 CMyString& CMyString::operator+=(const CMyString& rhs) 59 { 60 // TODO: 여기에 return 문을 삽입합니다. 61 } </pre>

■ 과제 6 : CMyString.cpp 파일에 추가한 CMyString operator+(const CMyString& rhs), CMyString& operator+=(const CMyString& rhs) 연산자 함수의 코드 완성(Append() 함수를 사용)

StringCtrlSample.cpp 의 코드를 다음과 같이 수정

```
#include <iostream>
#include "CMyString.h"
using namespace std;
int main()
{
    CMyString strLeft("학번 : 0000"), strRight(", 이름 : 가나다"), strResult; // 학번과 이름은 본인 것으로 수정

    strResult = strLeft + strRight;
    cout << strResult << endl;

    cout << strLeft << endl;

    strLeft += strRight;
    cout << strLeft << endl;

    return 0;
}
```

실행결과가 다음과 같이 나오는지 확인!

```
CMyString 이동생성자 호출
학번 : 0000, 이름 : 가나다
학번 : 0000
학번 : 0000, 이름 : 가나다
계속하려면 아무 키나 누르십시오 . . .
```

실습과제#7 : 문자열은 배열이다

CMyString 클래스에도 배열 연산자가 꼭 필요합니다. 본디 ‘문자열’이라는 것이 배열이기 때문이죠.

다음과 같은 사용자 코드 및 실행 결과를 확인할 수 있도록 CMyString 클래스에 두 가지 배열 연산자 함수를 추가합니다.

CMyString.h	CMyString.cpp
<pre>class CMyString { public: CMyString(); ~CMyString(); // 변환생성자 explicit CMyString(const char* szParam); // 복사생성자 CMyString(const CMyString& rhs); // 이동생성자 CMyString(CMyString&& rhs); int GetLength() const; char& operator[](int nIndex); char operator[](int nIndex) const; CMyString operator+(const CMyString& rhs); CMyString& operator+=(const CMyString& rhs); CMyString& operator=(const CMyString& rhs);</pre>	<pre>53 char& CMyString::operator[](int nIndex) 54 { 55 // TODO: 여기에 return 문을 삽입합니다. 56 } 57 char CMyString::operator[](int nIndex) const 58 { 59 } 60 }</pre>

- 과제 7 : CMyString.cpp 파일에 추가한 char& operator[](int nIndex), char operator[](int nIndex) const; 연산자 함수의 코드를 완성해보세요~

```

#include <iostream>
#include "CMyString.h"
using namespace std;

void TestFunc(const CMyString& strParam)
{
    for (int i = 0; i < strParam.GetLength(); i++)
    {
        cout << strParam[i] << " ";
    }
    cout << endl;
}

int main()
{
    CMyString strParam("StudentID : 0000, Name : 가나다"); // 학번과 이름은 본인 것으로 수정

    cout << strParam.GetString() << endl;

    TestFunc(strParam);

    return 0;
}

```

실행결과가 다음과 같이 나오는지 확인!

```

StudentID : 0000, Name : 가나다
S t u d e n t I D   :   0 0 0 0 ,   N a m e   :   ???????
계속하려면 아무 키나 누르십시오 . . .

```

실습과제#8 : 문자열과 관계연산자

실습과제 8은 CMyString 클래스에 == 와 != 라는 관계 연산자 함수를 만드는 일입니다.

■ == 연산자함수, != 연산자함수 추가

CMyString.h	CMyString.cpp
<pre>class CMyString { public: CMyString(); ~CMyString(); // 변환생성자 explicit CMyString(const char* szParam); // 복사생성자 CMyString(const CMyString& rhs); // 이동생성자 CMyString(CMyString&& rhs); int GetLength() const; int operator==(const CMyString& rhs); int operator!=(const CMyString& rhs);</pre>	<pre>53 int CMyString::operator==(const CMyString& rhs) 54 { 55 return 0; 56 } 57 58 int CMyString::operator!=(const CMyString& rhs) 59 { 60 return 0; 61 } 62</pre>

■ == 연산자함수, != 연산자함수 정의

```
53  int CMyString::operator==(const CMyString& rhs)
54  {
55      if (m_pszData != NULL && rhs.m_pszData != NULL)
56      {
57          if (strcmp(m_pszData, rhs.m_pszData) == 0)
58              return 1;
59      }
60
61      return 0;
62  }
63
64  int CMyString::operator!=(const CMyString& rhs)
65  {
66      if (m_pszData != NULL && rhs.m_pszData != NULL)
67      {
68          if (strcmp(m_pszData, rhs.m_pszData) == 0)
69              return 0;
70      }
71      return 1;
72  }
```

```

#include <iostream>
#include "CMyString.h"
using namespace std;
int main()
{
    CMyString strParam("StudentID : 0000"), strName("Name : 가나다"); // 학번,이름은 본인 것으로 수정

    if (strParam == strName)
        cout << "같다! " << endl;
    else
        cout << "다르다! " << endl;

    CMyString strTest = CMyString("StudentID : 0000"); // 학번은 본인 것으로 수정

    if (strParam != strTest)
        cout << "다르다!" << endl;
    else
        cout << "같다! " << endl;

    return 0;
}

```

실행결과가 다음과 같이 나오는지 확인!

```

다르다!
같다!
계속하려면 아무 키나 누르십시오 . . .

```

고생했습니다~!