문자열

c++ 문자열 – string 클래스

- string 클래스 -> c++ 문자열처리는 string 클래스 사용을 권고
  - C++ 표준 라이브러리, <string> 헤더 파일에 선언
  - 가변 크기의 문자열
  - 다양한 문자열 연산을 실행하는 연산자와 멤버 함수 포함
  - 문자열, 스트링, 문자열 객체, string 객체 등으로 혼용
  - null 문자로 끝나지 않음

- 문자열 생성

```
string str;    // 빈 문자열을 가진 스트링 객체
string address1("강원도 춘천시 한림대학길");
string address2("강원도 춘천시 한림대학길", 6);  //address2="강원도";
string copyAddress(address1);

getline(cin, str, 'c'); //getline() 문자열 입력 함수, 세번째 인수는 구분 기호(생략하면 ₩n), 문자 'c'에서 입력 중지

char text[] = {'L', 'o', 'v', 'e', ' ', 'C', '+', '+', '₩0'}; // C-스트링(char [] 배열)으로부터 스트링 객체 생성
string title(text);

auto string1="Hello cpp";  //const char*
auto string2="Hello cpp"s;  //str::string
```

c++ 문자열 – string 클래스

- string 객체의 동적 생성과 소멸
  - new/delete를 이용하여 문자열을 동적 생성/반환 가능

```
string *p = new string("C++");  // 스트링 객체 동적 생성
cout << *p; // "C++" 출력

p->append(" Great!!"); // p가 가리키는 스트링이 "C++ Great!!"이 됨
cout << *p; // "C++ Great!!" 출력

delete p; // 스트링 객체 반환
```

- 숫자 <-> 문자열 변환

```
//숫자-> 문자열로 변환
 cout<<to_string(130)<<endl;
 cout<<to_string(3.14)<<endl;

 //문자열 -> 숫자로 변환
 cout<<stoi("1234aa")<<endl; //int, 오류 아님, 1234로 변환
 cout<<stol("3456")<<endl; //long, stol("a3456") -> error
 cout<<stod("12.34")<<endl; //double
 cout<<stof("34.45")<<endl; //float
```

문자열 다루기(1)

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string s1 = "C++ programming", s2="language";
    cout << "문자열 비교 : " << s1.compare(s2) << endl;
    cout << "문자열 연결 : " << s1.append(s2) << endl;

    s1="C++ programming";

    cout << "문자열 삽입 : " << s1.insert(3, "really") << endl;
    cout << "문자열 대치 : " << s1.replace(3, 6, "study") << endl; //3부터 6개의 문자를 study로 대치
    cout << "문자열 길이 : " << s1.length() << endl;  cout << "문자열 길이 : " << s1.size() << endl;
    cout << "문자열 삭제 : " << s1.erase(0, 4) << endl; //문자열 일부분 삭제0부터 4개 문자 삭제

    s2 = s1; //문자열 복사, 깊은 복사
    s1.clear();  //문자열 전체 삭제
    cout << "문자열 삭제 s1 : " << s1 << endl;
    cout << "문자열 복사 후 s2 : " << s2 << endl;
    return 0;
}
```

```
문자열 비교 : -1
문자열 연결 : C++ programminglanguage
문자열 삽입 : C++really programming
문자열 대치 : C++study programming
문자열 길이 : 20
문자열 길이 : 20
문자열 삭제 : tudy programming
문자열 삭제 s1 :
문자열 복사 후 s2 : tudy programming
```

문자열 다루기 (2)

```
#include <iostream>
#include <string>
#include <locale> //문자를 다루는 함수, isdigit(), toupper(), isalpha(), islower()
using namespace std;
int main() {
    string s1 = "C++ programming";

    cout << "문자열 검색 : " << s1.find("p") << endl; //문자나 문자열 검색, 없으면 -1반환
    cout << "문자열 추출 : " << s1.substr(2,4) << endl; //문자열 일부 추출, 인덱스 2에서 4개의 문자 반환
    cout << "문자열 추출 : " << s1.substr(2) << endl; //문자열 일부 추출, 인덱스 2에서 끝까지 반환
    cout << "문자열의 각 문자 다루기 : " << s1.at(5) << endl; //인덱스 5에 있는 문자 반환
    cout << "문자 다루기 : " << static_cast<char>(toupper('a')) << endl;

    if (isdigit(s1.at(6)))  cout << "숫자" << endl;
    else if (isalpha(s1.at(6))) cout << "문자" << endl;
    else  cout<<"해당없음"<<endl;
    return 0;
}
```

```
문자열 검색 : 4
문자열 추출 : + pr
문자열 추출 : + programming
문자열의 각 문자 다루기 : r
문자 다루기 : A
문자
```

string_view 클래스 – c++ 17

- string_view 클래스
  - 다양한 타입의 문자열 처리
  - 임시 객체 생성없이 문자열 사용
  - 단, 언어 표준이 C++17인 경우만 동작

```
#include <string_view> //string_view 클래스

//string_view 클래스 사용
string_view extractExt(string_view filename) {
    return filename.substr(filename.find('.'));
}
int main() {
    string filename = "C:₩temp₩file.string";
    //data() : string_view 타입 문자열 -> string 타입 문자열 생성
    //cout << extractExt(filename).data()<< endl;
    cout << extractExt(filename)<< endl;

    const char* cfilename = "C:₩temp₩file.cstring";
    cout << extractExt(cfilename)<< endl;

    cout << extractExt("C:₩temp₩file.literal")<< endl;
    return 0;
}
```

```
PS C:\yanges\lecture\lecture_src\cpp> g++ shared.cpp -std=c++17
PS C:\yanges\lecture\lecture_src\cpp> ./a
.string
.cstring
.literal
```

```
#pragma warning(disable : 4996)
#include <iostream>
#include <string>
#include <cstring>
using namespace std;
```

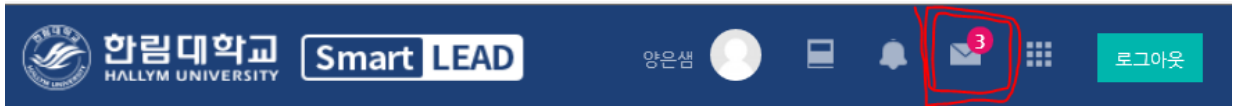
```
.string
.cstring
```

```
//다양한 타입의 문자열 처리를 위해 함수 오버로딩
string extractExt(const string& filename) {
    return filename.substr(filename.find('.'));
}

char* extractExt(char* filename) {
    char *p = nullptr;
    char ext[10];
    p = strchr(filename, '.');
    strcpy(ext, p+1);
    return ext;
}

int main() {
    string filename = "C:₩temp₩file.string";
    cout << extractExt(filename)<< endl;

    const char* cfilename = "C:₩temp₩file.cstring";
    cout << extractExt(cfilename)<< endl;

    cout << extractExt("C:₩temp₩file.literal")<< endl;
    return 0;
}
```

Q & A

- "C++ 문자열"에 대한 학습이 모두 끝났습니다.

- 새로운 내용이 많았습니다. 모든 내용을 이해 하셨나요?
- 아직 이해가 안되는 내용이 있다면 다시 한번 복습하시기 바랍니다.
- 질문은 한림 SmartLEAD 쪽지 또는 e-mail 또는 전화상담을 이용하시기 바랍니다.



- cpp_05_문자열_ex.pdf 에 확인 학습 문제들을 담았습니다.
- 이론 학습을 완료한 후 확인 학습 문제들로 학습 내용을 점검 하시기 바랍니다.

- 퀴즈와 과제가 출제되었습니다. 마감시간에 늦지 않도록 주의해 주세요.

- 수고하셨습니다.^^