C++ 프로그래밍 실습

Visual Studio 2015

Contents

- Type Casts
- String
- References
- Exercise

- **Practice1 –Type Casts**

Practice 1 : Type Casts

- Static cast (**`static_cast`**)
- Constant cast (**`const_cast`**)
- Reinterpret cast (**`reinterpret_cast`**)
- Dynamic cast (**`dynamic_cast`**)

Practice 1-1 : Static cast

static_cast<바꾸려고 하는 타입>(대상);

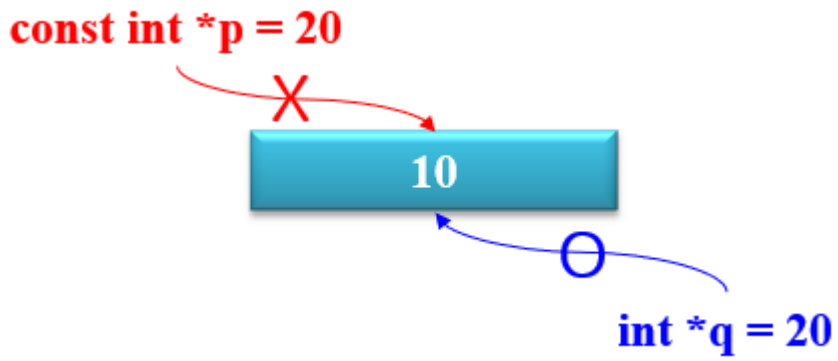
```
#include<iostream>
using namespace std;
int main() {
   int int_val = 123;
   float float_val;

•      float_val = static_cast<float>(int_val); //int ->float로 형변환
   cout << "static_cast<float>(int_val) : " << float_val << endl;
}
```

Execution Result:

```
static_cast<float>(int_val) : 123
```

Practice 1-2 : Constant cast

```
#include<iostream>

using namespace std;

int main() {
    const int*p = new int(10);
    int *q = const_cast<int*>(p);
    *q = 200;
    cout << *p << endl;
}

//포인터 상수성(const) 제거
```



Execution Result:

200

Practice 1-3 :Reinterpret cast

```
#include<iostream>
using namespace std;

int main() {
   int num = 0x010203;
   char *ptr = reinterpret_cast<char*>(&num);
//int형 정수에 바이트 단위 접근을 위해서 int형 포인터를 char형 포인터로 형 변환

   for (int i = 0; i < sizeof(num); i++) {
      cout << static_cast<int>(*(ptr + i)) << endl;
//바이트 단위 데이터를 문자가 아닌 정수의형태로 출력하기 위해서 char형 데이터를 int형 으로 변환
   }
}
```

- **Practice2 – String**

Practice 2 : String

- Data type

Type	Typical Bit Width	Typical Range
bool	1byte	true, false
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	Range	0 to 65,535
signed short int	Range	-32768 to 32767
long int	4bytes	-2,147,483,647 to 2,147,483,647
signed long int	4bytes	same as long int
unsigned long int	4bytes	0 to 4,294,967,295
float	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	8bytes	+/- 1.7e +/- 308 (~15 digits)
wchar_t	2 or 4 bytes	1 wide character

Practice 2-1 : String

```
string s1 = “Atlas”, s2 = “King”, s3;
s3 = s1 + ‘ ’ + s2;
cout << s1 << '\n';
cout << s2 << '\n';
cout << s3 << '\n';
s1 += s2;
cout << s1 << '\n';
cout << s2 << '\n';
```

```
Atlas
King
Atlas King
AtlasKing
King
```

Practice 2-1 : String

```
#include<iostream>
#include<string>
using namespace std;

int main() {
    string s1 = "건국대학교";
    string s2 = "C++";
    string s3 = s1 +' '+ s2;
    string s4 = s3 + "수업";

cout << "s1 : " << s1 << endl;
cout << "s2 : " << s2 << endl;
cout << "s3 : " << s3 << endl;
cout << "s4 : " << s4 << endl;

}
```

Execution Result:

```
s1 : 건국대학교
s2 : C++
s3 : 건국대학교 C++
s4 : 건국대학교 C++ 수업
```

Practice 2-2 : String

Removing a substring from the string

```
string s = "Ray Dennis Steckler";
s.erase(4, 7);
cout << s << '\n';
```

```
Ray Steckler
```

Inserting a string at a given position

```
string s1 = "Ray Steckler", s2 = "Dennis ";
s1.insert(4, s2);
cout << s1 << '\n';
```

```
Ray Dennis Steckler
```

Replacing a substring with a given string

```
string s1 = "Ray Dennis Steckler", s2 = "Fran";
s1.replace (4, 6, s2);
cout << s1 << '\n';
```

```
Ray Fran Steckler
```

Swapping two strings

```
string s1 = "Ray Dennis Steckler", s2 = "Ed Wood";
s1.swap(s2);
cout << s1 << '\n' << s2 << '\n';
```

```
Ed Wood
Ray Dennis Steckler
```

Practice 2-2 String

```
#include<iostream>
#include<string>
using namespace std;
int main() {
    string s1 = "C++ Programming Practice";
    string s2 = "Programming";
    s1.erase(4, 12);
    cout << s1 << endl;
    s1.insert(4, s2);
    cout << s1 << endl;
    s1.replace(4, 11, "Programming ");
    cout << s1 << endl;

    int i;
    i = s1.length();
    cout << "Length : " << i << endl;
}
```

Execution Result:

```
C++ Practice
C++ ProgrammingPractice
C++ Programming Practice
24
```

Practice 2-3 : String

Referencing a char at a specified index like an array

```
string s = "Nan";
cout << s[1] << '\n';
s[0] = 'J';
cout << s << '\n';
```

```
a
Jan
```

Extracting a substring

```
string s1 = "Ray Dennis Steckler";
string s2;
s2 = s1.substr(4, 6);
cout << s1 << '\n';
cout << s2 << '\n';
```

```
Ray Dennis Steckler
Dennis
```

- Search a string (1)
  - **s1.find(s2, *ind*)**
    - Searches string **s1** for a substring **s2** from the beginning of **s1**
    - If **s2** is found, returns the smallest index ≥ ***ind*** where **s2** begins, if **s2** is not found, "plus infinity"
    - Note that **s1.find(s2)** = s1.find(s2, 0)
  - **s1.rfind(s2, *ind*)**
    - Searches string **s1** for a substring **s2** from the end of **s1** (reversely)
    - If **s2** is found, returns the largest index ≤ ***ind*** where **s2** begins, if **s2** is not found, "plus infinity"
    - Note that **s1.rfind(s2)** = s1.rfind(s2, s1.length()-1)

```
string s1 = "Ray Dennis Dennis Steckler";
string s2 = "Dennis";
int f = s1.find(s2);
int r = s1.rfind(s2);
cout << "Found at index: " << f << '\n';
cout << "Found at index: " << r << '\n';
```

```
Found at index: 4
Found at index: 11
```

Practice 2-3 String

```
#include<iostream>
#include<string>
using namespace std;
int main() {
string s1 = "C++ Programming Practice";
string s2;

cout << s1[1] << s1[0] << endl;

s2 = s1.substr(4, 11);
cout << s2 << endl;
}
```

Execution Result:

```
+C
Programming
```

```
int main() {
   string s1 = "C++ Programming Practice Programming";
   string s2 = "Programming";

   int f = s1.find(s2);
   int l = s1.rfind(s2);

   cout << "Found at index " << f << endl;
   cout << "Found at index " << l << endl;
}
```

Execution Result:

```
Found at index 4
Found at index 25
```

noskipws & getline()

getline() 사용 형식
getline(stream, string)

```
#include <iostream>
using namespace std;

void main() {
    char c;
    cin >> noskipws;
    while (cin >> c)cout << c;
}
```

cin.getline() 사용 형식
cin.getline(변수, 최대입력가능 문자수,종결문자)

```
#include<iostream>
#include<string>

using namespace std;

int main() {
string address;
char address2[100];
cout << "주소를 입력하세요 " << endl;
getline(cin, address);  //string

cout << "주소를 입력하세요 " << endl;
cin.getline(address2, 100); //char

cout << "입력한 주소(str)는 : " << address << endl;
cout << "입력한 주소(char)는 : " << address2 << endl;

}
```

- **Practice3 – References**

Practice 3-1 Reference

```
#include<iostream>
using namespace std;

int main() {
    int num1 = 1020;
    int &num2 = num1; //참조자 선언 -> num1 과 num2 동일한 메모리 공간 참조

    num2 = 3047;
    cout << "VAL: " << num1 << endl;
    cout << "REF: " << num2 << endl;

    cout << "VAL: " << &num1 << endl;
    cout << "REF: " << &num2 << endl;
}
```

Execution Result:

```
VAL: 3047
REF: 3047
VAL: 00FFF8D8
REF: 00FFF8D8
```

Practice 3-2 : Call by Reference

With "***call by reference***", parameters of a function refer to the actual arguments passed to the function

By default, copies of the arguments are passed as "***call by value***"

So far in C, we have implemented "*call by reference*" with pointers

From now in C++, we can implement "*call by reference*" with references (no dereferencing required)

```
void main() {
    int i = 7, j = -3;
    swap(i, j);
    cout << "i = " << i << '\n'
         << "j = " << j << '\n';
}

// Call by value version of swap
// : Why does this func. fail?
void swap(int a, int b) {
    int t;
    t = a; a = b; b = t;
}
```

```
// Pointer version of swap
void swap(int* a, int* b) {
    int t;
    t = *a; *a = *b; *b = t;
}

// Reference version of swap
void swap(int& a, int& b) {
    int t;
    t = a; a = b; b = t;
    // No dereferencing
}
```

Practice 3-2 Call by Reference

```
#include<iostream>

using namespace std;

int main() {
    int i = 7, j = 3;
    cout << "i = " << i << endl;
    cout << "j = " << j << endl;
    swap(i, j);
    cout << "i = " << i << endl;
    cout << "j = " << j << endl;
}
void swap(int&a, int&b) {
    int t;
    t = a; a = b;  b = t;
}
```

Execution Result:

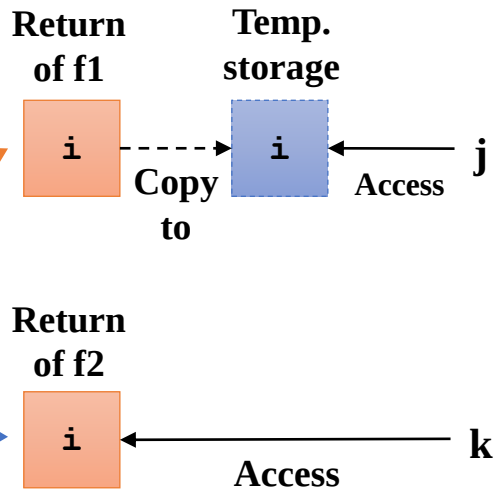
```
i = 7
j = 3
i = 3
j = 7
```

Practice 3-3 : Return by Reference

- ***Return by reference***
  - The return value is not copied, rather the actual storage is made available to the invoking function

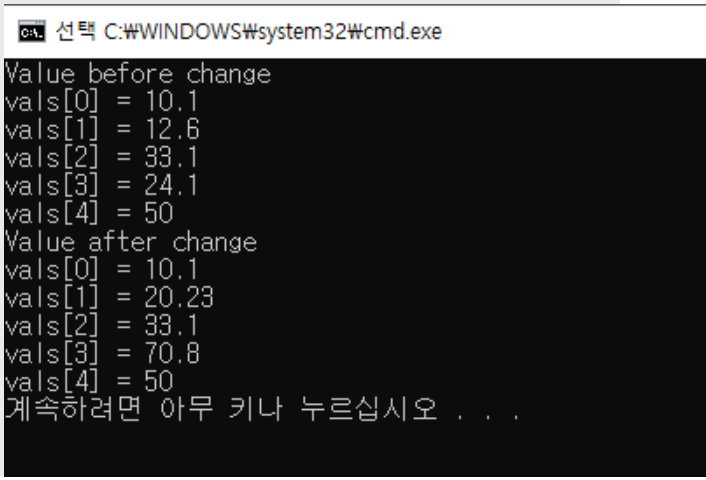
```
int f1()    // return by value
{
    ...
   return i;
}

int& f2()  // return by reference
{
    ...
   return i;
}
```

```
void main()
{
    ...
   int j;
   j = f1();

    ...
   int k;
   k = f2();
}
```



Practice 3-3 Return by reference

```
#include <iostream>

using namespace std;

double vals[] = { 10.1, 12.6, 33.1, 24.1, 50.0 };

double& setValues(int i)
{
return vals[i];
}

int main() {

cout << "Value before change" << endl;
for (int i = 0; i < 5; i++) {
cout << "vals[" << i << "] = ";
cout << vals[i] << endl;
}

setValues(1) = 20.23;
setValues(3) = 70.8;

cout << "Value after change" << endl;
for (int i = 0; i < 5; i++) {
cout << "vals[" << i << "] = ";
cout << vals[i] << endl;
}
return 0;
}
```



Exercise-1

string s1 = "**If you don't walk today, "**
string s2 = "**you will have to run tomorrow"**
를 string 함수를 사용하여

"**If you do not walk today, you will have to run tomorrow tomorrow**"
이렇게 만들고 Call by Reference를 이용하여

"you will have to run tomorrow tomorrow If you do not walk today,"

위와 같은 결과값을 출력하시오.

출력 : "**If you do not walk today, you will must run tomorrow**"
"you will must run tomorrow, If you do not walk today"

Course Homepage

- How to access
  - URL: sclab.konkuk.ac.kr
- Downloading class material
  - Students can download syllabus and lecture notes in PDF format
- Class announcement
  - About homework and project
  - Exam schedule and result
  - And so on

Submit

- Teaching assistant: 장성수
  Office: 신공학관 1216호 (대학원 SCLab 연구실)
  Email: pik1100@naver.com

- Title of the email : [2018][Practice#]_student# _ student _ name

- Ex) [2018][Practice02]_201700000_장성수

- Create zip file. (C++ project folder)

- 주의 : 메일 양식이 잘못될 경우 채점이 되지 않을 수 있음.

- 질문 메일 : pik1100@naver.com : 장성수

끝