10강. 복사 생성자

C++ 프로그래밍

jhhwang@kumoh.ac.kr

컴퓨터공학과 황준하

목차

복사 생성의 의미

- 변수의 생성 및 복사 생성
  - int a = 3; → 일반 생성
  - int b = a; → 복사 생성 : 변수 생성 시 다른 변수를 토대로 생성
    - int b(a); // int b = a; 와 동일한 문법
- 객체의 생성 및 복사 생성

비교 : CPoint(3, 4);

```
void main(void)
{
    CPoint P1(3, 4);
    CPoint P2(P1);
    CPoint P3 = P2;
}
```

일반 생성

이것이 복사 생성이다.

복사 생성 시에는
해당 클래스의
복사생성자가 호출됨!

복사 생성이 발생하는 경우

- 복사 생성이 발생하는 경우
  - 원칙 : 새로운 객체가 생성될 때 기존 객체(1개)를 기반으로 만듦
  - CPoint P2(P1); → 복사 생성
  - CPoint P3(P1, 3); → 일반 생성

- 복사 생성이 발생하는 경우의 예
  - 객체의 값에 의한 매개변수 전달
  - 객체의 값에 의한 반환

복사 생성이 발생하는 경우의 예 (1)

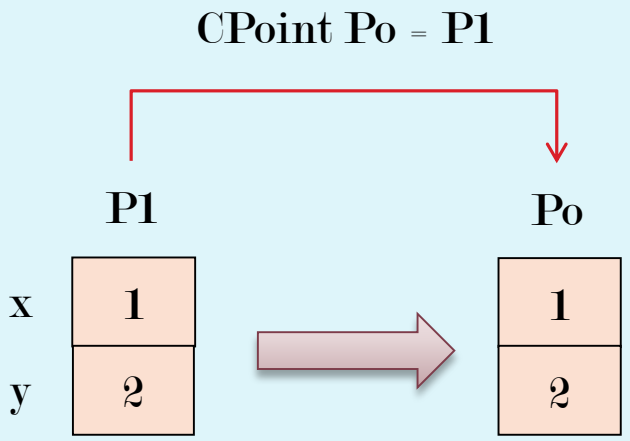
- 객체의 값에 의한 매개변수 전달

```
class CPoint {
private :
      int x, y;

public :
      CPoint(int a = 0, int b = 0) : x(a), y(b) { }
      void Print() { cout << "(" << x << ", " << y << ")" << endl; }
};

void ShowPoint(CPoint Po)
{
      Po.Print();
}

void main(void)
{
      CPoint P1(1, 2);
      ShowPoint(P1);
}
```

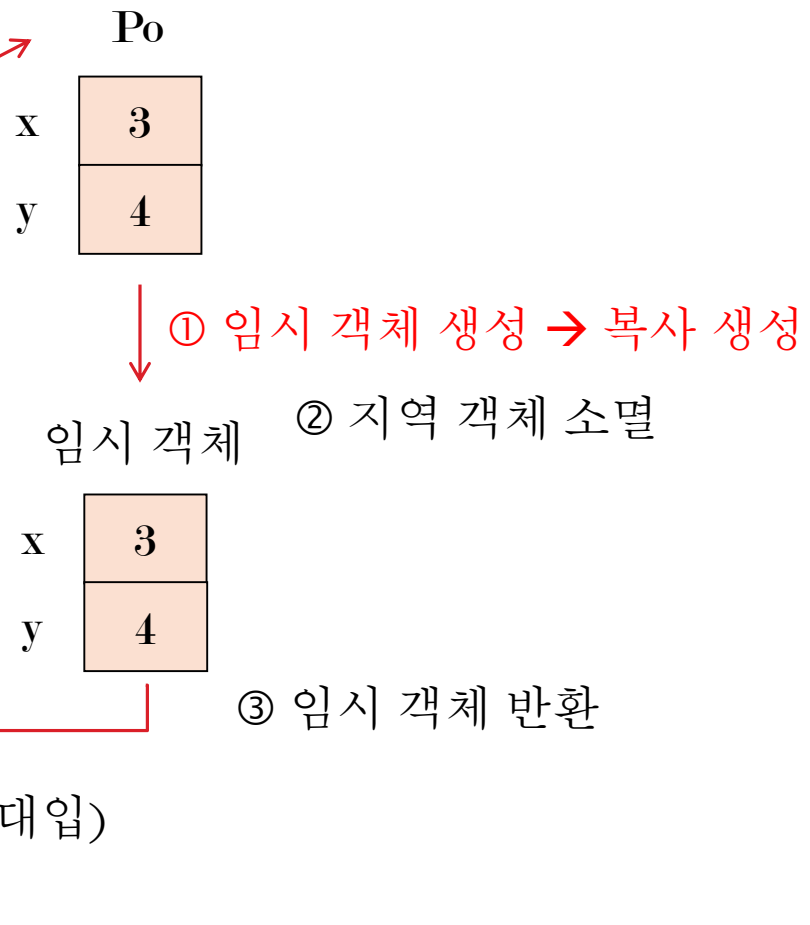


복사 생성이 발생하는 경우의 예 (2)

- 객체의 값에 의한 반환

```
CPoint GetPoint(void)
{
    CPoint Po(3, 4);
    return Po;
}

void main(void)
{
    CPoint P1;
    P1 = GetPoint();
    P1.Print();
}
```



디폴트 복사 생성자

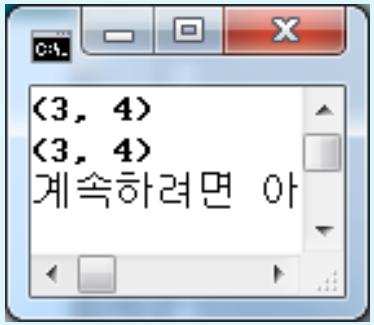
- 다음 프로그램은 정상적으로 동작할까?

```
class CPoint {
private :
	int x, y;

public :
	CPoint(int a = 0, int b = 0) : x(a), y(b) { }
	void Print() { cout << "(" << x << ", " << y << ")" << endl; }
};

void main(void)
{
	CPoint P1(3, 4);
	CPoint P2 = P1;

	P1.Print();
	P2.Print();
}
```

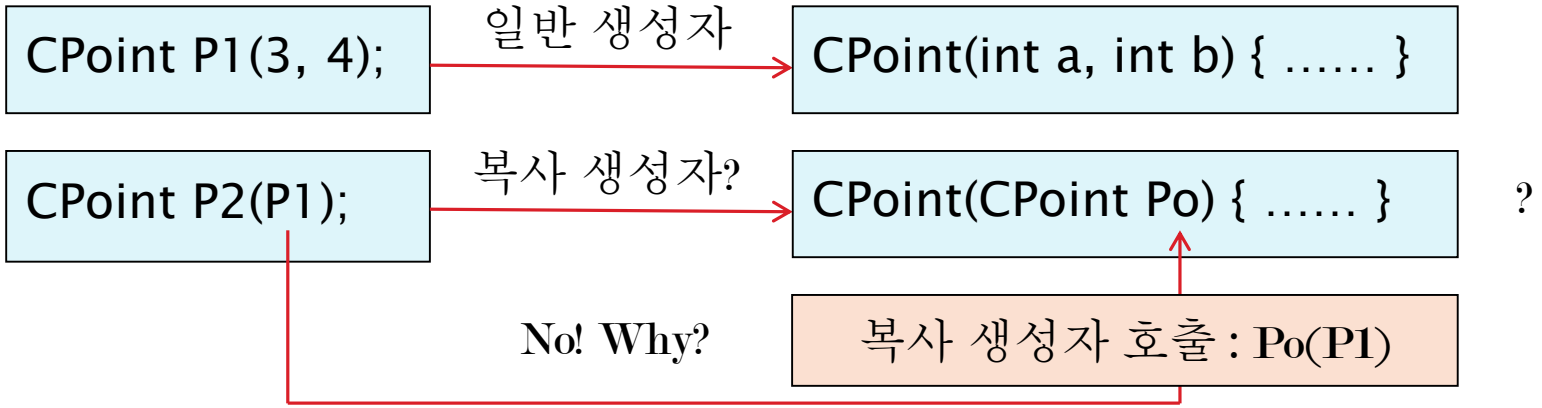


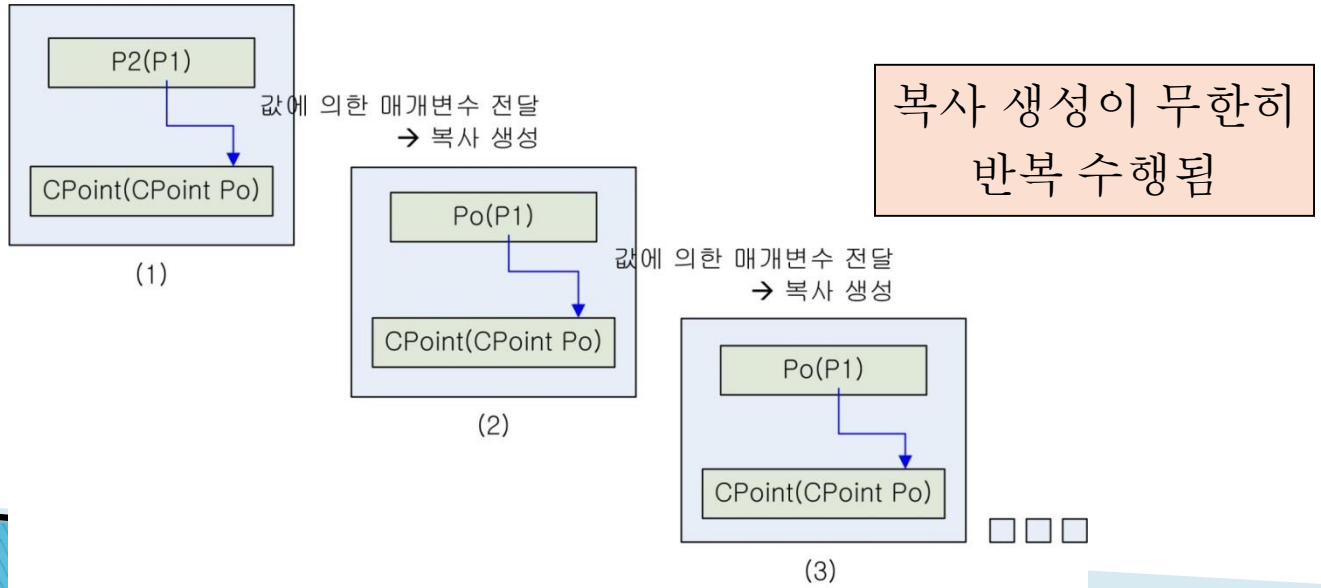
복사 생성

복사 생성자는 어디에? → 디폴트 복사 생성자 존재

디폴트 복사 생성자

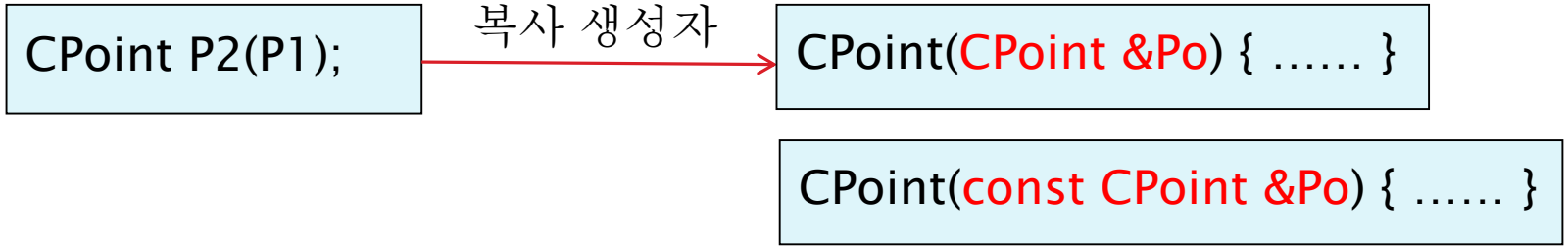
- 복사 생성자는 어떻게 생겼을까?





디폴트 복사 생성자

- 복사 생성자의 모양



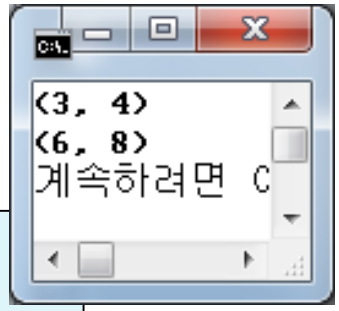
- 디폴트 복사 생성자
  - 멤버 단위 복사 : 멤버 변수 단위로 대입

```
class CPoint {
private :
      int x, y;

public :
      CPoint(const CPoint &Po) : x(Po.x), y(Po.y) { }
};
```

복사 생성자의 재정의

- 디폴트 복사 생성자만으로 충분한가?
  - No!
  - 필요한 경우 복사 생성자를 재정의해야 함
- 복사 생성자를 명시적으로 재정의할 경우
  - 디폴트 복사 생성자 사라짐
  - 디폴트 생성자 역시 사라짐




```
void main(void)
{
        CPoint P1(3, 4);
        CPoint P2 = P1;
        P1.Print();
        P2.Print();
}
```

```
class CPoint {
private :
        int x, y;

public :
        CPoint(const CPoint &Po) { x = Po.x * 2; y = Po.y * 2; }
        CPoint(int a = 0, int b = 0) : x(a), y(b) { }
        void Print() { cout << "(" << x << ", " << y << ")" << endl; }
};
```

CString 클래스의 복사 생성자

- 다음 프로그램의 문제점은?

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

class CString {
private :
    int len;
    char *str;

public :
    CString(char *s = "Unknown") {
        len = strlen(s);
        str = new char[len + 1];
        strcpy(str, s);
    }
    ~CString() { delete [] str; }
    void Print() { cout << str << endl; }
};
```

문자열의 길이

문자열

```
void main(void)
{
    CString str1 = "C++ Programming";
    CString str2 = str1;

    str1.Print();
    str2.Print();
}
```

복사 생성

생성자 : 문자열 생성

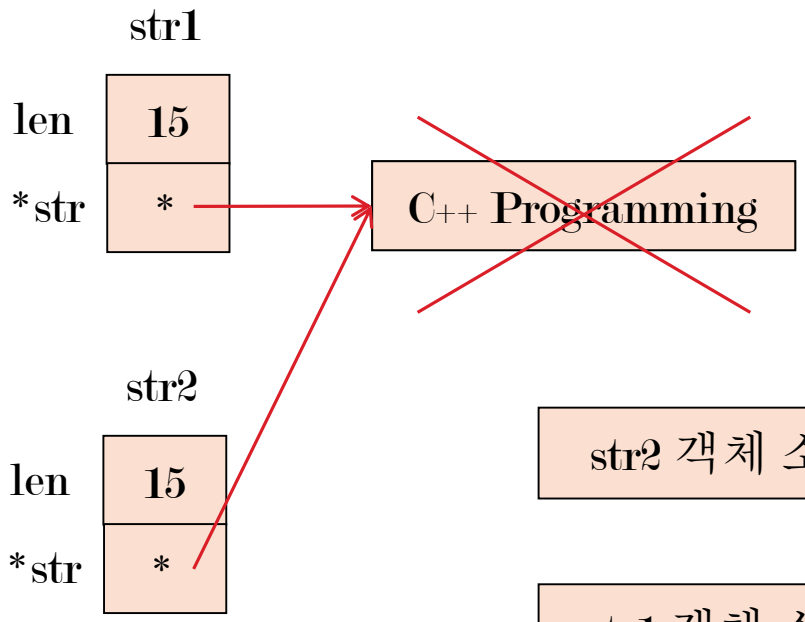
소멸자 : 문자열 메모리 해제

CString 클래스의 복사 생성자

- 문제점 분석

```
void main(void)
{
        CString str1 = "C++ Programming";
        CString str2 = str1;

        str1.Print();
        str2.Print();
}
```

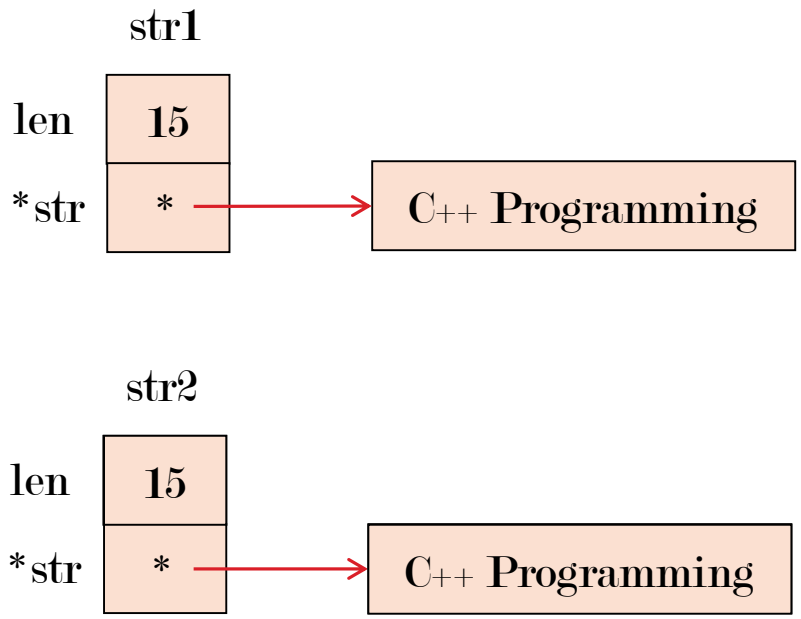


str2 객체 소멸 → 소멸자 → delete [] str

str1 객체 소멸 → 소멸자 → delete [] str → 에러!

CString 클래스의 복사 생성자

- 올바른 동작




```
void main(void)
{
	CString str1 = "C++ Programming";
	CString str2 = str1;

	str1.Print();
	str2.Print();
}
```

1. len값 복사 (대입)
2. (len+1)바이트만큼 메모리 확보
3. 문자열 복사

str1, str2 객체에 대한 소멸자가 각각 수행되더라도 문제 없음!

CString 클래스의 복사 생성자

- CString 클래스의 복사 생성자

```
class CString {
private :
        int len;
        char *str;

public :
        CString(const CString &S) {
                len = S.len;
                str = new char[len + 1];
                strcpy(str, S.str);
        }
        // 나머지 생략
};
```

```
void main(void)
{
        CString str1 = "C++ Programming";
        CString str2 = str1;

        str1.Print();
        str2.Print();
}
```

1. len값 복사 (대입)
2. (len+1)바이트만큼 메모리 확보
3. 문자열 복사

복사 생성자 정리

- 복사 생성자
  - 디폴트 복사 생성자 : 멤버 단위 복사
  - 필요한 경우 복사 생성자를 명시적으로 재정의
- 복사 생성자의 모양
  - ClassName(const ClassName &Arg) { ..... }
- 복사 생성자를 재정의한 경우
  - 디폴트 복사 생성자 사라짐
  - 디폴트 생성자 사라짐

연습 문제

- 다음과 같이 CArray 클래스를 만들고 main 함수와 같이 사용하였다. 제대로 수행될 수 있도록 프로그램을 완성하라.

```
void main(void)
{
	CArray Ary1(5);
	CArray Ary2 = Ary1;

	Ary1.Print();
	Ary2.Print();
}
```

```
class CArray {
private :
	int count;
	int *ary;
public :
	CArray(int co = 1) {
		count = co; ary = new int[count];
		for (int i = 0; i < count; i++) ary[i] = i;
	}
	~CArray() { delete [] ary; }
	void Print() {
		for (int i = 0; i < count; i++)cout << ary[i] << " ";
		cout << endl;
	}
};
```

연습 문제

- 프로그램 확인
  - 복사 생성자 추가

```
CArray(const CArray &Ary) {
        count = Ary.count;
        ary = new int[count];
        for (int i = 0; i < count; i++)
                ary[i] = Ary.ary[i];
}
```

1. count 원소 크기 복사 (대입)
2. count 만큼 메모리 확보
3. 배열 복사