5장. 복사 생성자

담당교수 : 김미경
부산대학교

5-1 C++ & C 스타일 초기화

- 두 가지 형태의 초기화
  - 1_초기화1.cpp, 2_초기화2.cpp

```cpp
/*  1_초기화1.cpp*/
#include<iostream>
using std::cout;
using std::endl;

int main(void)
{
    int val1(20);
    int val2=40;

    cout<<"val1: "<<val1<<endl;
    cout<<"val2: "<<val2<<endl;

    return 0;
}
```

```cpp
/*  2_초기화2.cpp*/
#include<iostream>
using std::cout;using std::endl;
class AAA{
    int val;
public:
    AAA(int n){
        val=n;
    }
    void ShowData(){
        cout<<val<<endl;
    }
};

int main(void){
    AAA a1(10);
    a1.ShowData();
    AAA a2=20;
    a2.ShowData();
    return 0;
}
```

5-2 복사 생성자의 형태

```
/*3_복사생성자1.cpp*/
#include<iostream>
using std::cout; using std::endl;

class AAA
{
public:
    AAA(){
        cout<<"AAA() 호출"<<endl;
    }
    AAA(int i){
        cout<<"AAA(int i) 호출"<<endl;
    }
    AAA(const AAA& a){
        cout<<"AAA(const AAA& a) 호출"<<endl;
    }
};

int main(void) {
    AAA obj1;
    AAA obj2(10);
    AAA obj3(obj2);
    return 0;
}
```

5-3 디폴트 복사 생성자

- 디폴트 복사 생성자
  - 사용자 정의 복사 생성자가 없을 때 자동 삽입
  - 멤버 변수 대 멤버 변수의 복사를 수행
  - 4_디폴트복사생성자.cpp, 5_복사생성자3.cpp
- 디폴트 복사 생성자 복사 형태
  - 얕은 복사(Shallow Copy)!

5-3 디폴트 복사 생성자

```cpp
/*4_디 폴 트 복 사 생 성 자 .cpp*/
#include<iostream>
using std::cout; using std::endl;

class Point
{
    int x, y;
public:
    Point(int _x, int _y){
        x=_x;
        y=_y;
    }
    void ShowData(){
        cout<<x<<' '<<y<<endl;
    }
};
int main(void)
{
    Point p1(10, 20);
    Point p2(p1);

    p1.ShowData();
    p2.ShowData();
    return 0;
}
```

```cpp
/*5_복사생성자3.cpp*/
#include<iostream>
using std::cout;using std::endl;

class Point{
    int x, y;
public:
    Point(int _x, int _y){
        x=_x;
        y=_y;
    }
    Point(const Point& p){
        x=p.x;
        y=p.y;
    }
    void ShowData(){
        cout<<x<<' '<<y<<endl;
    }
};

int main(void){
    Point p1(10, 20);
    Point p2(p1);

    p1.ShowData();
    p2.ShowData();
}
```

5-4 Deep Copy

- 디폴트 복사 생성자의 문제점
  - 얕은 복사에 의한 메모리 참조 오류!
  - 6_얕은복사생성자.cpp, 7_깊은복사생성자.cpp

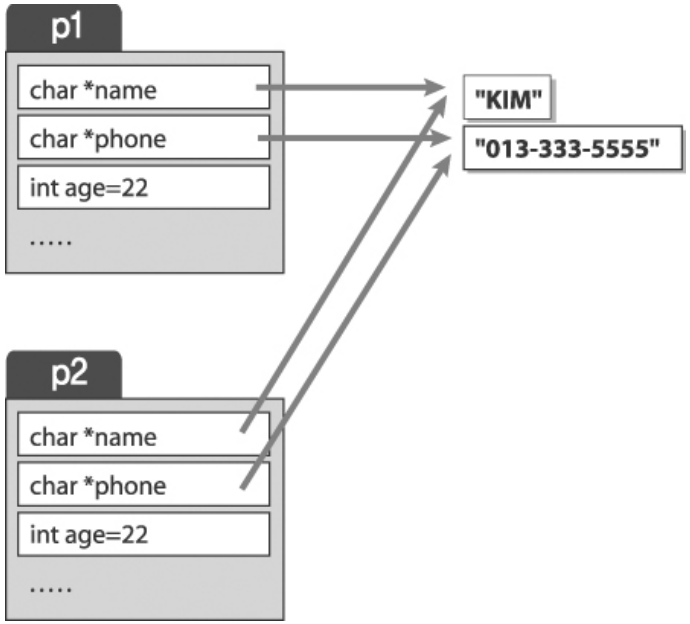

그림 5-4

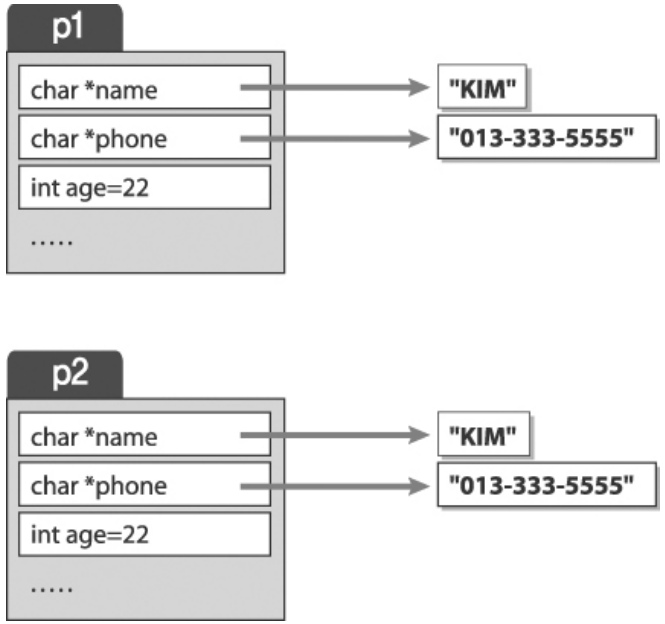

그림 5-5

5-4 Deep Copy

```
/* 6_얕은복사생성자.cpp
   Person.cpp에서 정의한 Person 클래스*/
#include<iostream>
#include <string.h>
using std::cout; using std::endl;

class Person {
    char *name;
    char *phone;
    int age;
public:
    Person(char* _name, char* _phone, int _age);
    ~Person();
    void ShowData();
};
Person::Person(char* _name, char* _phone, int _age)
    name=new char[strlen(_name)+1];
    strcpy(name, _name);

    phone=new char[strlen(_phone)+1];
    strcpy(phone, _phone);

    age=_age;
}
Person::~Person(){
    delete []name;
    delete []phone;
}
void Person::ShowData(){
    cout<<"name: "<<name<<endl;
    cout<<"phone: "<<phone<<endl;
    cout<<"age: "<<age<<endl;
}

int main()
{
    Person p1("KIM", "013-333-5555", 22);
    Person p2=p1;
    p1.ShowData() ;
    p2.ShowData();

    return 0;
}
```

5-4 Deep Copy

```cpp
/* 7_깊은복사생성자.cpp*/
#include<iostream>
using std::cout;  using std::endl;

class Person
{
    char *name;
    char *phone;
    int age;
public:
    Person(char* _name, char* _phone, int _age);
    Person(const Person& p);
    ~Person();
    void ShowData();
};
Person::Person(const Person& p){
    name=new char[strlen(p.name)+1];
    strcpy(name, p.name);

    phone=new char[strlen(p.phone)+1];
    strcpy(phone, p.phone);

    age=p.age;
}
```

```cpp
Person::Person(char* _name, char* _phone, int _age){
    name=new char[strlen(_name)+1];
    strcpy(name, _name);

    phone=new char[strlen(_phone)+1];
    strcpy(phone, _phone);

    age=_age;
}
Person::~Person() {
    delete []name;
    delete []phone;
}
void Person::ShowData(){
    cout<<"name: "<<name<<endl;
    cout<<"phone: "<<phone<<endl;
    cout<<"age: "<<age<<endl;
}

int main(){
    Person p1("KIM", "013-333-5555", 22);
    Person p2=p1;
    p1.ShowData();
    p2.ShowData();
}
```

5-5 복사 생성자의 호출 시기

- 복사 생성자 호출 형태 3가지
  - Case 1
    - 기존에 생성된 객체로 새로운 객체 초기화
  - Case 2
    - 함수 호출 시 객체를 값에 의해 전달
  - Case 3
    - 함수 내에서 객체를 값에 의해 리턴

5-5 복사 생성자의 호출 시기

- **Case 1**

```
class AAA
{
    int val;
public:
    AAA(int i){
        val=i;
    }
    AAA(const AAA& a){
        cout<<"AAA(const A& a) 호출"<<endl;
        val=a.val;
    }
    void ShowData(){
        cout<<"val: "<<val<<endl;
    }
};
```

```
int main()
{
    AAA obj1(10);
    AAA obj2=obj1;
    return 0;
}
```

5-5 복사 생성자의 호출 시기

- **Case 2**

```
void function(AAA a) {
    a.ShowData();
}

int main()
{
    AAA obj(30);
    function(obj);
    return 0;
}
```

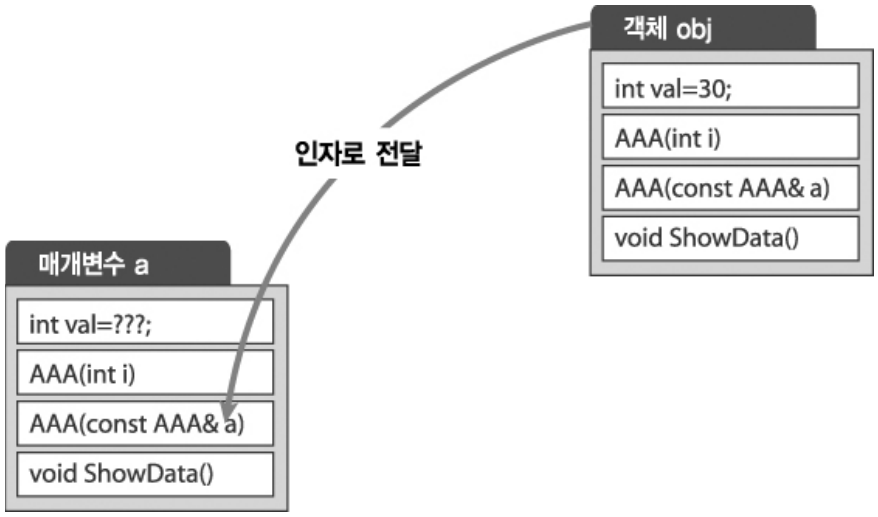

그림 5-8

5-5 복사 생성자의 호출 시기

- **Case 3**

```
AAA function(void)
{
    AAA a(10);
    return a;
}

int main()
{
    function();
    function().ShowData();
   return 0;
}
```

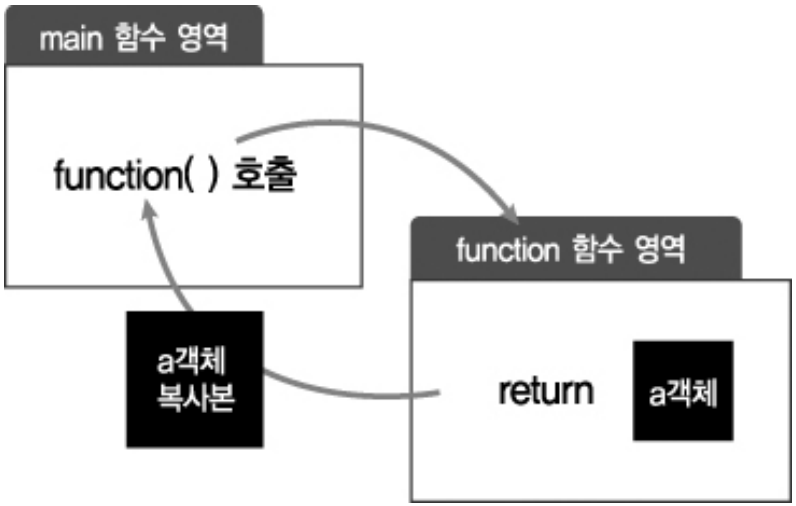

그림 5-11

5-5 복사 생성자의 호출 시기

```cpp
/* 9_복사생성자_경우2.cpp*/
#include<iostream>
using std::cout; using std::endl;

class AAA {
    int val;
public:
    AAA(int i){
        cout<<"AAA(int i)생성자 호출"<<endl;
        val=i;
    }
    AAA(const AAA& a){
        cout<<"AAA(const A& a) 호출"<<endl;
        val=a.val;
    }
    void ShowData(){
        cout<<"val: "<<val<<endl;
    }
};
void function(AAA a){
    a.ShowData();
}
int main() {
    AAA obj(30);
    function(obj);
}
```

```cpp
/*10_복사생성자_경우3*/
#include<iostream>
using std::cout;    using std::endl;
class AAA{
    int val;
public:
    AAA(int i=0){
        cout<<"AAA(int i)생성자 호출"<<endl;
        val=i;
    }
    AAA(const AAA& a){
        cout<<"AAA(const A& a) 호출"<<endl;
        val=a.val;
    }
    void ShowData(){
        cout<<"val: "<<val<<endl;
    }
};
AAA function(void){
    cout << "function 안 출력 " <<endl;
    AAA a(10);
    cout << "function에서 return 전 " <<endl;
    return a;
}
int main(){
    AAA b=function();
    return 0;
}
```

```
function 안 출력
AAA(int i)생성자 호출
function에서 return 전
AAA(const A& a) 호출
```