
5장. 복사 생성자

담당교수 : 김미경
부산대학교

5-1 C++ & C 스타일 초기화

■ 두 가지 형태의 초기화

- 1_초기화1.cpp, 2_초기화2.cpp

```
1  /*  1_초기화1.cpp */
2  #include<iostream>
3  using std::cout;
4  using std::endl;
5
6  int main(void)
7  {
8      int val1(20);
9      int val2=40;
10
11      cout<<"val1: "<<val1<<endl;
12      cout<<"val2: "<<val2<<endl;
13
14      return 0;
15 }
```

```
1  /*  2_초기화2.cpp */
2  #include<iostream>
3  using std::cout;using std::endl;
4  class AAA{
5      int val;
6  public:
7      AAA(int n){
8          val=n;
9      }
10     void ShowData(){
11         cout<<val<<endl;
12     }
13 };
14
15 int main(void){
16     AAA a1(10);
17     a1.ShowData();
18     AAA a2=20;
19     a2.ShowData();
20     return 0;
21 }
```

5-2 복사 생성자의 형태

```
1  /*3_복사생성자1.cpp*/
2  #include<iostream>
3  using std::cout; using std::endl;
4
5  class AAA
6  {
7  public:
8      AAA(){
9          cout<<"AAA() 호출"<<endl;
10     }
11     AAA(int i){
12         cout<<"AAA(int i) 호출"<<endl;
13     }
14     AAA(const AAA& a){
15         cout<<"AAA(const AAA& a) 호출"<<endl;
16     }
17 };
18
19 int main(void) {
20     AAA obj1;
21     AAA obj2(10);
22     AAA obj3(obj2);
23     return 0;
24 }
```

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
 - 얇은 복사[Shallow Copy]

5-3 디폴트 복사 생성자

```
1  /*4_디폴트 복사 생성자 .cpp*/
2  #include<iostream>
3  using std::cout; using std::endl;
4
5  class Point
6  {
7      int x, y;
8  public:
9      Point(int _x, int _y){
10         x=_x;
11         y=_y;
12     }
13     void ShowData(){
14         cout<<x<<' '<<y<<endl;
15     }
16 };
17 int main(void)
18 {
19     Point p1(10, 20);
20     Point p2(p1);
21
22     p1.ShowData();
23     p2.ShowData();
24     return 0;
25 }
```

```
1  /*5_복사생성자3.cpp*/
2  #include<iostream>
3  using std::cout;using std::endl;
4
5  class Point{
6      int x, y;
7  public:
8      Point(int _x, int _y){
9          x=_x;
10         y=_y;
11     }
12     Point(const Point& p){
13         x=p.x;
14         y=p.y;
15     }
16     void ShowData(){
17         cout<<x<<' '<<y<<endl;
18     }
19 };
20
21 int main(void){
22     Point p1(10, 20);
23     Point p2(p1);
24
25     p1.ShowData();
26     p2.ShowData();
27 }
```

5-4 Deep Copy

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

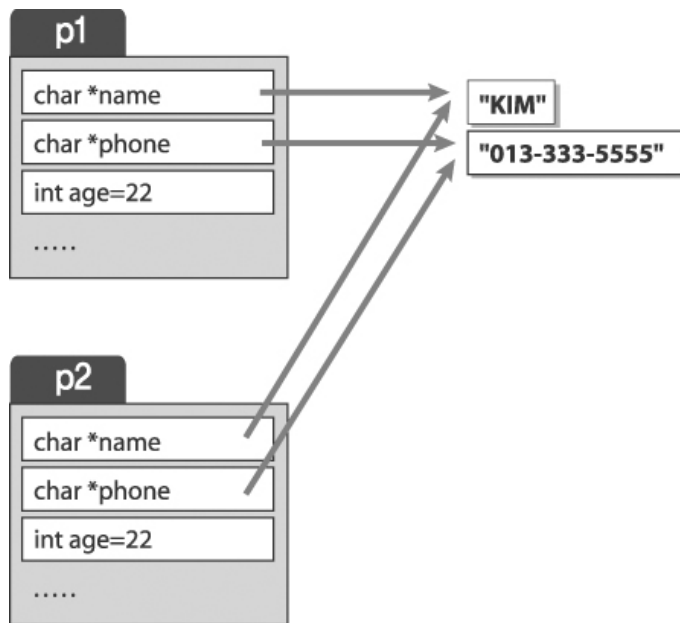


그림 5-4

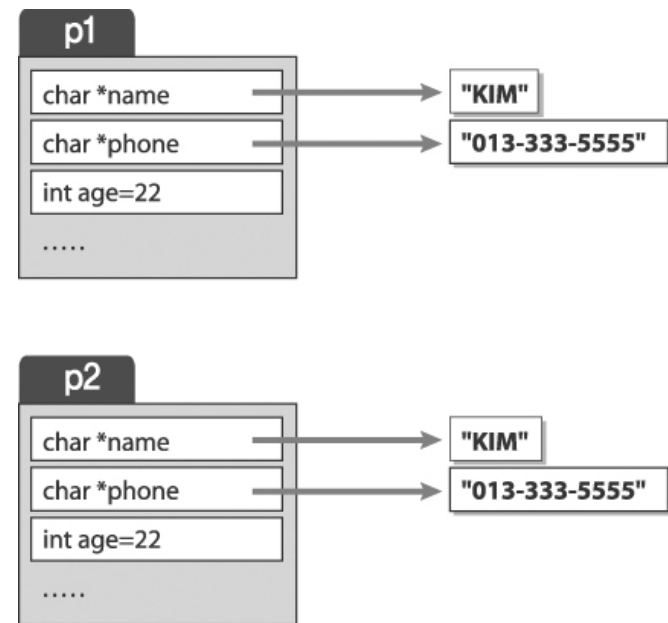


그림 5-5

5-4 Deep Copy

```
1  /* 6_얕은복사생성자.cpp
2     Person.cpp에서 정의한 Person 클래스*/
3  #include<iostream>
4  #include <string.h>
5  using std::cout; using std::endl;
6
7  class Person {
8      char *name;
9      char *phone;
10     int age;
11 public:
12     Person(char* _name, char* _phone, int _age);
13     ~Person();
14     void ShowData();
15 };
16 Person::Person(char* _name, char* _phone, int _age){
17     name=new char[strlen(_name)+1];
18     strcpy(name, _name);
19
20     phone=new char[strlen(_phone)+1];
21     strcpy(phone, _phone);
22
23     age=_age;
24 }
25 Person::~Person(){
26     delete []name;
27     delete []phone;
28 }
29 void Person::ShowData(){
30     cout<<"name: "<<name<<endl;
31     cout<<"phone: "<<phone<<endl;
32     cout<<"age: "<<age<<endl;
33 }
34
35 int main()
36 {
37     Person p1("KIM", "013-333-5555", 22);
38     Person p2=p1;
39     p1.ShowData() ;
40     p2.ShowData();
41
42     return 0;
43 }
```

5-4 Deep Copy

```
1  /* 7_깊은 복사 생성자 .cpp */
2  #include<iostream>
3  using std::cout; using std::endl;
4
5  class Person
6  {
7      char *name;
8      char *phone;
9      int age;
10 public:
11     Person(char* _name, char* _phone, int _age);
12     Person(const Person& p);
13     ~Person();
14     void ShowData();
15 };
16 Person::Person(const Person& p){
17     name=new char[strlen(p.name)+1];
18     strcpy(name, p.name);
19
20     phone=new char[strlen(p.phone)+1];
21     strcpy(phone, p.phone);
22
23     age=p.age;
24 }
25
```

```
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;  
}
```

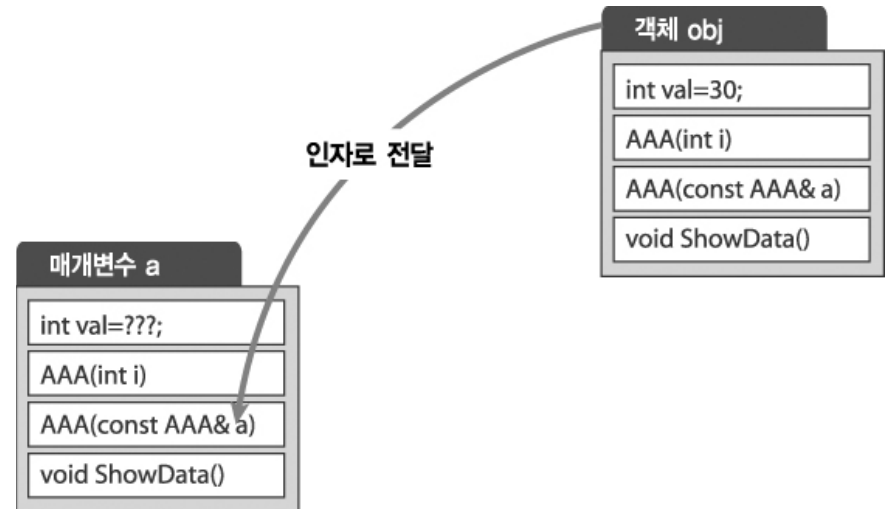


그림 5-8

5-5 복사 생성자의 호출 시기

■ Case 3

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

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

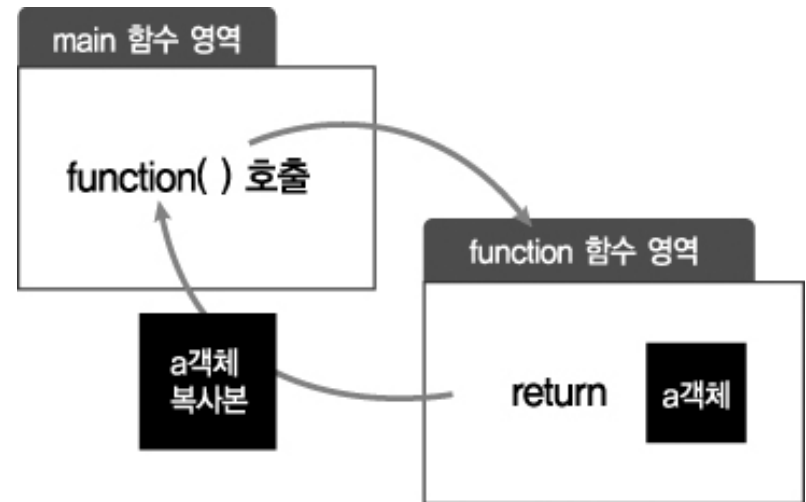


그림 5-11

5-5 복사 생성자의 호출 시기

```
1  /* 9_복사생성자_경우2.cpp */
2  #include<iostream>
3  using std::cout; using std::endl;
4
5  class AAA {
6      int val;
7  public:
8      AAA(int i){
9          cout<<"AAA(int i)생성자 호출"<<endl;
10         val=i;
11     }
12     AAA(const AAA& a){
13         cout<<"AAA(const A& a) 호출"<<endl;
14         val=a.val;
15     }
16     void ShowData(){
17         cout<<"val: "<<val<<endl;
18     }
19 };
20 void function(AAA a){
21     a.ShowData();
22 }
23 int main() {
24     AAA obj(30);
25     function(obj);
26 }
```

```
1  /*10_복사생성자_경우3*/
2  #include<iostream>
3  using std::cout; using std::endl;
4  class AAA{
5      int val;
6  public:
7      AAA(int i=0){
8          cout<<"AAA(int i) 생성자 호출"<<endl;
9          val=i;
10     }
11     AAA(const AAA& a){
12         cout<<"AAA(const A& a) 호출"<<endl;
13         val=a.val;
14     }
15     void ShowData(){
16         cout<<"val: "<<val<<endl;
17     }
18 };
19 AAA function(void){
20     cout << "function 안 출력 " <<endl;
21     AAA a(10);
22     cout << "function에서 return 전 " <<endl;
23     return a;
24 }
25 int main(){
26     AAA b=function();
27     return 0;
28 }
```

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