cpp_06_복사생성자

□ 개념 확인 학습

1. 다음 질문에 O, X로 답하세요.

A. 얕은 복사(shallow copy)는 객체 복사 시 객체의 멤버를 1:1대로 복사한다. ( )

B. 생성자와 소멸자의 비대칭 문제는 주소에 의한 호출로만 해결할 수 있다. ( )

C. 객체의 복사 생성 시 호출되는 특별한 생성자를 복사 생성자(copy constructor)라 한다. ( )

D. 복사 생성자는 클래스의 생성자처럼 한 클래스에 여러 개 선언이 가능하다. ( )

E. 자기 클래스에 대한 참조 매개변수를 가지는 생성자를 복사 생성자라 한다. ( )

F. 복사 생성자가 선언되어 있지 않는 클래스에 복사 생성자 호출이 필요한 경우가 발생할 경우, 컴파일러는 오류를 발생시킨다. ( )

G. 깊은 복사(deep copy)는 객체 복사 시 객체의 멤버변수에 동적메모리가 할당된 경우 원본이 가진 메모리의 크기만큼 사본에도 메모리를 별도로 동적 할당해야 한다. ( )

H. 깊은복사(deep copy)는 사본과 원본이 같은 공간의 메모리를 공유하게 된다. ( )

I. 변수처럼 이름과 주소를 가진 대상(지속되는 객체)을 Rvalue라고 한다. ( )

J. Rvalue reference를 사용할 때에는 &&를 사용한다.( )

K. 객체나 값을 전달할 때 복사를 사용하지 않고 소유권을 완전히 이동 시키는 것은 불가능하다. ( )

L. lvalue를 rvalue로 만들 때에는 move()를 사용한다. ( )

M. 이동 생성자와 이동 대입 연산자를 구현 할 때에는 Rvalue Reference를 파라미터로 받는 함수를 작성한다. ( )

2. 복사 생성자가 자동으로 호출되는 경우는 어떤 경우인지 설명하고 그 예를 보이세요.

3. Sample 객체 a에 대한 참조변수 ref를 선언하는 문장을 제시하세요.

4. Sample 클래스의 복사 생성자를 선언하는 함수원형을 제시하세요. 단, 매개변수는 s로 합니다.

5. Person 클래스의 선언이 다음과 같은 경우 Person 클래스의 복사 생성자를 선언하는 문장을 제시하세요.

```
class Person { //Person 클래스 선언
    char *name;
    int id;
public:
   Person(const Person& p); //복사 생성자
};
```

6. Person 클래스의 선언이 다음과 같은 경우 Person 클래스의 이동 생성자를 선언하는 문장을 제시하세요.

```
class Person { //Person 클래스 선언
    char *name;
    int id;
public:
   Person(Person&& p); //이동 생성자
};
```

□ 응용 프로그래밍

7. Accumulator 클래스는 생성자 매개변수로 전달된 값을 누적 저장하는 기능을 합니다. 아래에 제시된 클래스 선언과 main() 함수 그리고 실행결과를 참고하여 add() 함수를 작성하세요.

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

class Accumulator {
    int value;
public:
    Accumulator(int val) : value{val} { };
    Accumulator &add(int n);
    int get() { return value; }
};

//
//add() 함수를 작성합니다.
//

int main() {
    Accumulator acc(10);
    cout << acc.get() << endl;  //10 출력

    acc.add(1).add(2).add(3);  //acc 객체의 value는 16이 됨.
    cout << acc.get() << endl;  //16 출력
}
```

```
PS C:\yanges\lecture\lecture_src\cpp> g++ cpptest.cpp
PS C:\yanges\lecture\lecture_src\cpp> ./a
10
16
```

8. Account 클래스는 생성자 매개변수로 전달된 값을 멤버변수에 balance(잔액)에 저장합니다. 이 후 increaseBy() 함수에 객체와 입금액이 전달되면 전달된 객체의 balance(잔액)에 입금액을 누적 저장합니다. 아래에 제시된 클래스 선언과 main() 함수 그리고 실행결과를 참고하여 increaseBy() 함수를 작성하세요.

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

class Account {
   int balance; //잔액
public:
   Account(int deposit) : balance(deposit) { };
   int getBalance() { return balance; }
   void setBalance(int deposit) { this->balance = deposit; }
   void show() {
      cout << "잔액은 " << balance << "원 입니다." << endl;
   }
};

//
//increaseBy() 함수를 작성합니다.
//

int main() {
   Account acc(500);
   cout << "입금 전 "; acc.show();

   int in;
   cout << "입금액 : "; cin >> in;

   increaseBy(acc, in);
   cout << "입금 후 "; acc.show();
}
```

```
PS C:\yanges\lecture\lecture_src\cpp> g++ cpptest.cpp
PS C:\yanges\lecture\lecture_src\cpp> ./a
입금 전 잔액은 500원 입니다.
입금액 : 300
입금 후 잔액은 800원 입니다.
```

9. 아래에 제시된 클래스 선언과 main() 함수 그리고 실행결과를 참고하여 전달된 메시지를 출력하기 위한 message_print() 함수를 작성하세요.

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

//
//message_print() 함수를 구현합니다.
//

int main() {
   string stra = "apple";
   string strb = "banana";
   message_print(move(stra));
   message_print(stra + strb);
   return 0;
}
```

```
PS C:\yanges\lecture\lecture_src\cpp> g++ cpptest.cpp
PS C:\yanges\lecture\lecture_src\cpp> ./a
message = apple
message = applebanana
```

10. 아래에 제시된 클래스 선언과 main() 함수를 참고하여 복사 생성자와 이동 생성자를 구현한 후 출력결과를 예측하세요. 프로그램 수행 결과와 예측 결과를 비교하세요.

```
class Person {
    string name;
public:
    Person() = default;
    Person(string n) : name {n} { cout << "생성자 실행" << endl; };
    Person(const Person &person); //복사 생성자
    Person(Person&& p); //이동 생성자
    ~Person() { cout << "소멸자 실행" << endl; };
    void show(string obj) { cout << obj << " name = "<< name << endl; }
};

//
//복사 생성자와 이동 생성자를 구현하세요.
//

int main() {
    cout << "-1----------" << endl;
    Person dan("daniel");
    Person ben = Person("benny");

    cout << "-2----------" << endl;
    Person mvdan = move(dan);

    cout << "-3----------" << endl;
    Person cpben = ben;
    Person cpmvdan(mvdan);

    cout << "-4----------" << endl;
    dan.show("dan");
    ben.show("ben");
    mvdan.show("mvdan");
    cpben.show("cpben");
    cpmvdan.show("cpmvdan");
    return 0;
}
```

11. 위 10번 문제 Person class의 멤버변수 string name;을 char *name으로 변경하여 전체 프로그램을 수정하세요. 단 main() 함수의 변경은 없습니다.

```
class Person {
    char* name;
public:
    Person() = default;
    Person(const char *n);
    Person(const Person &person); //복사 생성자
    Person(Person&& p); //이동 생성자
    ~Person();
    void show(string obj);
};
```