

LAB #8



경희대학교
KYUNG HEE UNIVERSITY

● 실습 목표 – 상속과 집합의 이해 및 응용

1. 순수 가상 함수
2. 추상 클래스
3. 객체 형 변환
 - 객체 업캐스팅(Upcasting)
 - 포인터를 객체로 형 변환
4. 다중 상속의 개념 및 활용
5. 집합(Aggregation)

● 실습 문제 1 < Section 12-6 >

목표 : 순수 가상 함수를 이해하고 사용해본다.

문제 : 클래스 Base와 Derived가 있고 Derived가 Base를 상속받는다고 할 경우, 다음의 요구사항을 만족할 수 있도록 프로그램을 완성하라.

- 클래스 Base:

- 1) 멤버변수로 숫자 두 개를 저장하는 m_Num1, m_Num2를 가짐
- 2) 순수 가상 함수인 setNum(int val1, int val2)과 printSum() 메소드 선언

- 클래스 Derived:

- 1) 클래스 Base를 상속
- 2) 멤버변수는 없음. 멤버 함수는 다음의 두 함수를 가짐.
 - 1) 두 수를 Base 클래스의 m_Num1과 m_Num2에 저장하는 setNum(int val1, int val2) 함수를 구현함
 - 2) 두 수의 합을 출력하는 printSum() 함수를 구현함

- main() 함수가 있는 코드는 다음 장과 같이 주어짐

● 실습 문제 1 < Section 12-6 >

// 클래스 Based와 클래스 Derived 선언 및 구현 코드 작성

```
int main()
{
    Base baseObj; /* 이 라인을 삽입하여 Base 클래스의 객체를 생성하려고 시도할 때
                  무슨 에러가 나는 지 관찰하고, 그 이유를 생각해 봄.
                  그리고, 이 라인을 제거하고 다시 수행해 봄 */

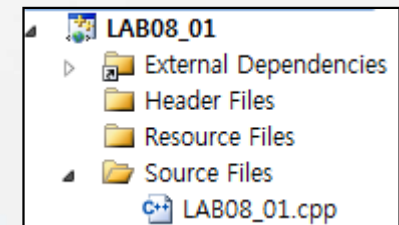
    Derived der;
    int num1, num2;

    cout << "두 수를 입력하세요 : " << endl;
    cin >> num1;
    cin >> num2;

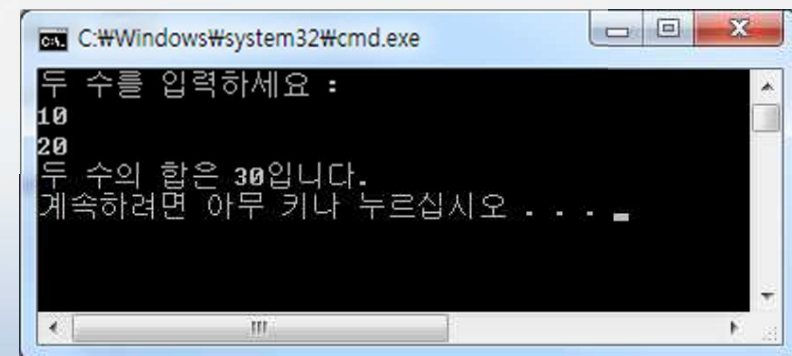
    // 클래스 B의 인스턴스를 통해
    // 두 수를 설정하고 합을 출력하는 코드 입력

    return 0;
}
```

프로젝트 구조



output



● 실습 문제 2 < Section 12-6 >

목표 : 순수 가상함수와 추상 클래스의 사용

문제 : Polygons 클래스를 상속받는 Triangle 클래스와 Rectangle 클래스를 구현한다.

- 추상클래스 Polygons:

- 1) 도형의 넓이(area)와 둘레(perimeter)의 두 멤버변수(protected) 를 가짐
- 2) 순수 가상 함수인 printArea() 메소드를 선언
- 3) 순수 가상 함수인 printPerimeter() 메소드를 선언

- 클래스 Triangle: Polygons 클래스 상속(public)

- 1) 가로(width)와 세로(height) 는 생성자의 초기화 리스트를 통해 초기화
- 2) calcArea() 메소드 구현 / 직각삼각형의 넓이를 계산함
- 3) calcPerimeter() 메소드 구현 / 직각삼각형의 둘레를 계산함

- 클래스 Rectangle: Polygons 클래스 상속(public)

- 1) 두 변의 길이(sideA, sideB)는 생성자의 초기화 리스트를 통해 초기화
- 2) calcArea() 메소드 구현 / 직사각형의 넓이를 계산함
- 3) calcPerimeter() 메소드 구현 / 직사각형의 둘레를 계산함

- main() 함수가 있는 코드는 다음 장과 같이 주어짐

● 실습 문제 2 < Section 12-6 >

```
int main()
{
    cout << "## 도형의 넓이와 둘레 구하기 ##" << endl;

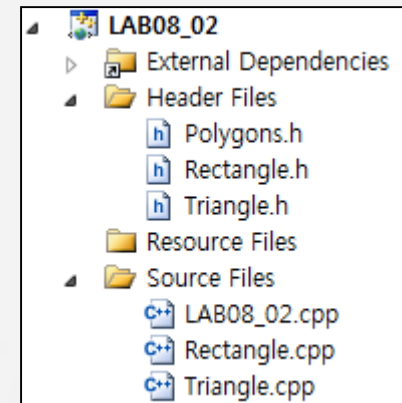
    Triangle tri(5, 12);
    // 직각삼각형의 넓이 및 둘레 구하는 코드 삽입

    Rectangle rect(10, 20);
    // 직사각형의 넓이 및 둘레 구하는 코드 삽입

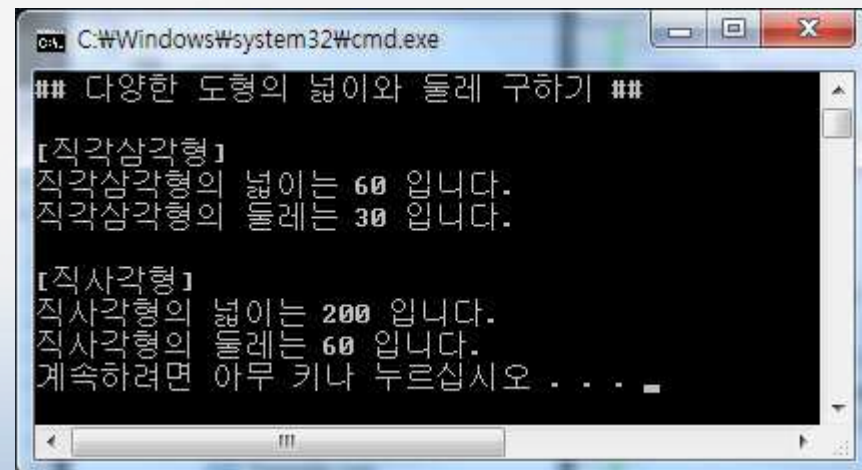
    // 직각삼각형의 넓이 및 둘레 출력하는 코드 삽입
    // 직사각형의 넓이 및 둘레 출력하는 코드 삽입

    return 0;
}
```

프로젝트 구조



output



● 실습 문제 3 < 12-7 >

목표 : 계층적 클래스에서의 형 변환(업캐스팅)

문제 : Derived 클래스는 Base 클래스를 상속받는 경우 다음 조건에 만족하는 프로그램을 완성하여라.

- 클래스 Base:

- 1) int형 변수 2개(m_val1, m_val2)를 가짐.
- 2) 두 수의 합을 출력하는 printSum()

- 클래스 Derived: **Base 클래스 상속(public)**

- 1) int형 변수 2개(m_val3, m_val4)를 가짐.
- 2) 생성자는 네 개의 수를 입력 받아 멤버를 초기화하는데, Base 클래스 멤버는 초기화 리스트를 통해 Base 생성자를 호출하여 초기화
- 3) 두 수의 합을 출력하는 printSum()

- main() 함수

- 1) Base와 Derived 인스턴스 생성
- 2) 두 인스턴스의 값 출력
- 3) **Derived 인스턴스를 Base 인스턴스에 대입(업캐스팅)**
- 4) 두 인스턴스의 값 출력(위의 출력과 비교)

• 실습 문제 3 < 12-7 >

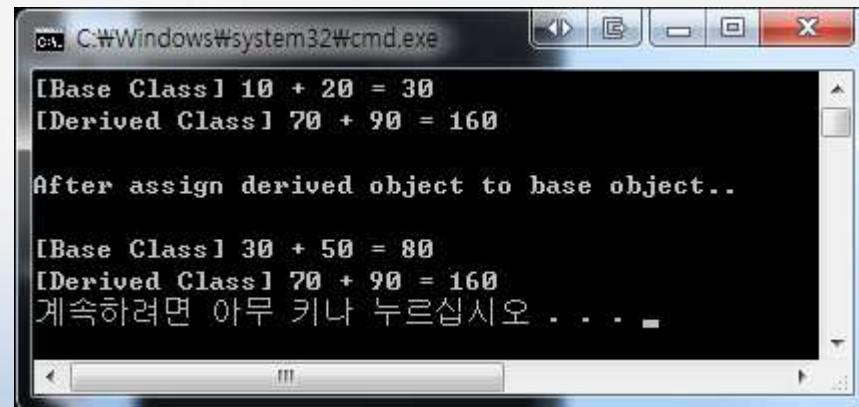
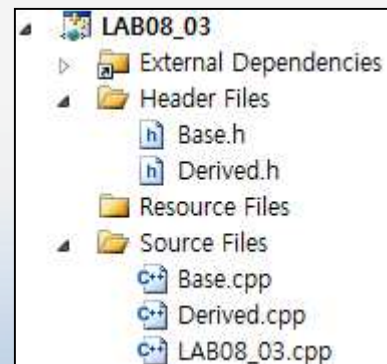
```
int main()
{
    Base baseObj(10, 20);
    Derived derObj(30, 50, 70, 90);

    baseObj.printSum();
    derObj.printSum();

    // Derived 클래스의 인스턴스를 Base 클래스 인스턴스에 대입 코드 작성(업캐스팅)
    derObj = baseObj; /* 이 라인을 삽입하여 baseObj로 derObj를 셋팅할 때는
                       무슨 에러가 나는 지 관찰하고, 그 이유를 생각해 봄.
                       그리고, 이 라인을 제거하고 다시 수행해 봄 */

    cout << endl << "After assign derived object to base object.." << endl << endl;
    baseObj.printSum();
    derObj.printSum();

    return 0;
}
```



● 실습 문제 4 < 12-7 >

목표 : 포인터를 객체로 형 변환(업캐스팅, 다운캐스팅)

문제 : 실습 3번의 구현을 다음의 요구사항으로 변경하여 완성하리라.

- 클래스 Base, Derived:

1) 실습 3번과 동일

- main() 함수

- 1) Base와 Derived 포인터 객체 생성
- 2) Derived 포인터를 Base 객체 형 변환(업캐스팅) 코드 작성
- 3) Base 포인터를 Derived 객체 형 변환(다운캐스팅) 코드 작성
- 4) 두 포인터 객체의 값 출력(위의 출력과 비교)

• 실습 문제 4 < 12-7 >

```
int main()
{
    cout << "[Upcasting Example]" << endl << endl;
    Base* basePtr = new Base(10, 20);
    Derived* derPtr = new Derived(30, 50);
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성
    // Derived 클래스의 객체 포인터를 Base 클래스 객체 포인터에 대입코드 작성(업캐스팅)
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성

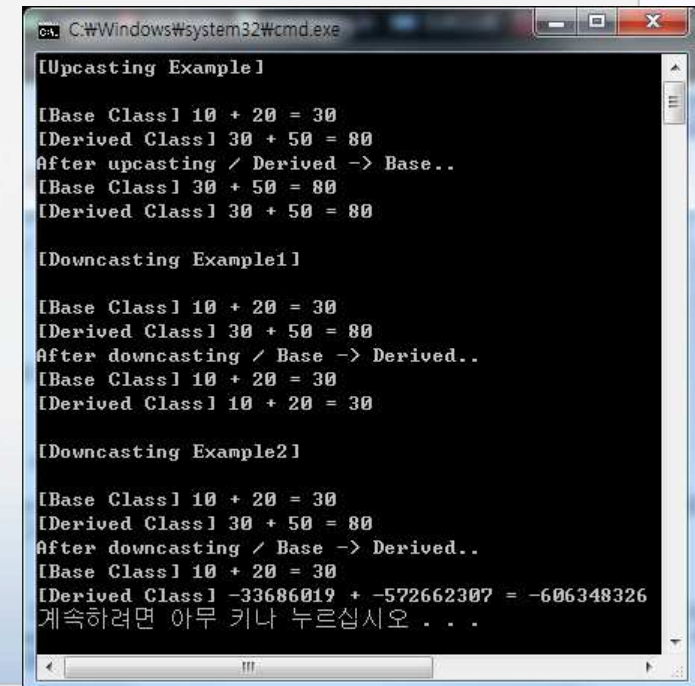
    cout << endl << "[Downcasting Example1]" << endl << endl;
    Base* basePtr2 = new Derived(10, 20);
    Derived* derPtr2 = new Derived(30, 50);
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성
    // Base를 Derived에 대입코드 작성(다운캐스팅)
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성

    cout << endl << "[Downcasting Example2]" << endl << endl;
    Base* basePtr3 = new Base(10, 20);
    Derived* derPtr3 = new Derived(30, 50);
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성
    // Base를 Derived에 대입코드 작성(다운캐스팅)
    // 두 객체에 대해, 각각 printSum()을 호출하는 코드 작성

    cout << "Example 2에서 다운캐스팅이 안되는 이유 작성"

    return 0;
}
```

output



```
C:\Windows\system32\cmd.exe
[Upcasting Example]

[Base Class] 10 + 20 = 30
[Derived Class] 30 + 50 = 80
After upcasting / Derived -> Base..
[Base Class] 30 + 50 = 80
[Derived Class] 30 + 50 = 80

[Downcasting Example1]

[Base Class] 10 + 20 = 30
[Derived Class] 30 + 50 = 80
After downcasting / Base -> Derived..
[Base Class] 10 + 20 = 30
[Derived Class] 10 + 20 = 30

[Downcasting Example2]

[Base Class] 10 + 20 = 30
[Derived Class] 30 + 50 = 80
After downcasting / Base -> Derived..
[Base Class] 10 + 20 = 30
[Derived Class] -33686019 + -572662307 = -606348326
계속하려면 아무 키나 누르십시오 . . .
```

● 실습 문제 5 < 12-8 >

목표 : 다중 상속의 개념 및 활용

문제 : 덧셈을 수행하는 Add 클래스와 뺄셈을 수행하는 Subtract 클래스를 선언하고, 이 두 클래스를 상속받는 Calculator 클래스를 구현하여라.

- 클래스 Add, Subtract:

- 1) 멤버변수: 연산 결과를 저장하는 double result;
- 2) 멤버함수
 - 두 수를 받아 계산한 후 결과를 result에 저장하는 생성자(Add, Subtract) 선언
 - 계산 결과를 출력하는 메소드(void add_result();, void subtract_result();)

- 클래스 Calculator: Add와 Subtract 다중상속(public)

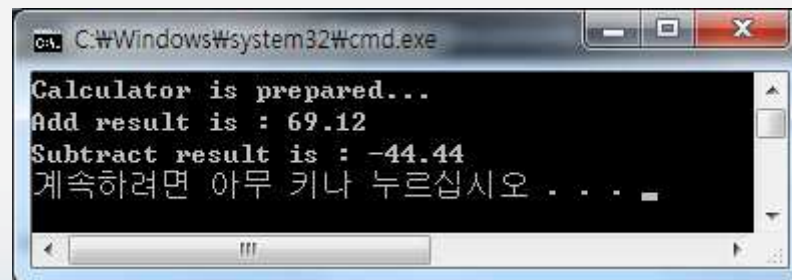
- 1) 멤버변수는 없음
- 2) 생성자에서 두 수를 받아 Add와 Subtract 클래스 생성자를 호출하여 초기화함(초기화리스트 사용)

• 실습 문제 5 < 12-8 >

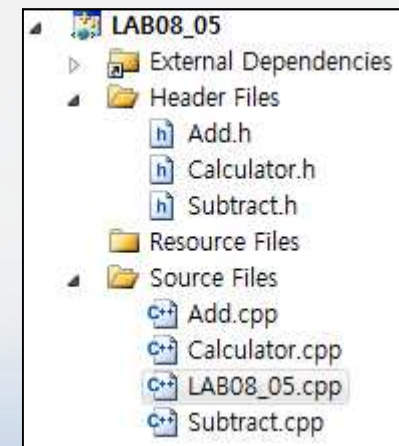
```
int main()
{
    Calculator calc(12.34, 56.78);
    calc.add_result();
    calc.sub_result();

    return 0;
}
```

output



프로젝트 구조



● 실습 문제 6 < 12-9 >

목표 : 클래스 내에서 집합(Aggregation)의 사용

문제 : 성적에 관한 클래스 Score와 학생에 관한 클래스 Student를 설계하고, Student 클래스에서 Score 클래스형 멤버변수를 갖는 프로그램을 작성하여라.

- 클래스 Score:

- 1) 멤버변수: 국어, 영어, 수학, 과학 성적을 저장하기 위한 int 형 변수 네 개
- 2) 멤버함수
 - 4과목 성적을 받아 초기화하는 생성자(초기화 리스트를 통해 멤버변수 초기화)
 - void setScore(int, int, int, int) : 4과목 성적을 받아 멤버변수 초기화
 - Score getScore() : 현재 Score 클래스를 리턴함.
 - void printScore(): 4과목 성적을 출력

- 클래스 Student

- 1) 멤버변수: 학생의 이름(char* m_name), 학번(int m_ID), 성적(Score m_score)
- 2) 멤버함수
 - 학생의 이름, 학번, 성적을 받아 초기화 하는 생성자(학번과 성적은 초기화 리스트 사용, 이름은 동적 배열을 할당 받아 문자열 복사)
 - void setVal(char*, int, Score) : 학생의 정보를 받아 멤버변수 초기화
 - void printVal() : 학생 정보 출력

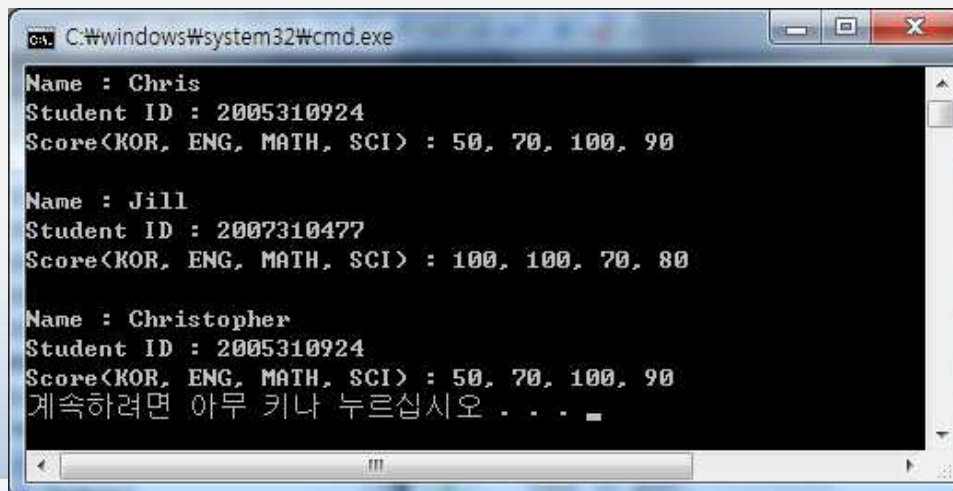
• 실습 문제 6 < 12-9 >

```
int main()
{
    Score chrisScore(50, 70, 100, 90);
    Score jillScore(100, 100, 70, 80);
    // getScore()를 사용하여 christopherScore를 chrisScore로 초기화 코드 작성

    Student std1("Chris", 2005310924, chrisScore);
    // 아래 출력결과를 참조하여 Jill과 Christopher에 대한 std2, std3 선언 및 값 대입 코드 작성
    // std3는 setVal()을 통해 초기화 코드 작성

    // 위의 세 학생의 정보 출력 코드 작성
    return 0;
}
```

output



```
C:\windows\system32\cmd.exe
Name : Chris
Student ID : 2005310924
Score<KOR, ENG, MATH, SCI> : 50, 70, 100, 90

Name : Jill
Student ID : 2007310477
Score<KOR, ENG, MATH, SCI> : 100, 100, 70, 80

Name : Christopher
Student ID : 2005310924
Score<KOR, ENG, MATH, SCI> : 50, 70, 100, 90
계속하려면 아무 키나 누르십시오 . . .
```

프로젝트 구조

