객체지향프로그래밍 (202004100 장주훈, 202084023 이병길)

**10주차** – 상속

**중요 포인트**! **업 캐스팅 다운 캐스팅과 객체 포인터**

Protected 접근 지정자

상속: 기본 클래스 / 파생클래스(기본 클래스에서 파생 클래스로 갈수록 클래스의 개념이 구체화)

* (부모클래스에서 자식클래스가 기능을 가져오는 느낌)

Class MobilePhone : public Phone{ //phone에 있는 내용을 mobliephone에서 상속받는다.

**텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명**

**상속 개념을 도입해 분리해보기!**

Class Student

Class StudentWorker : public Studen(Student를 상속받는 StudentWorker 선언)

Class Researcher

Class Professor : public Researcher(Researcher를 상속받는 Professor 선언)

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  // 2차원 평면에서 한 점을 표현하는 클래스 Point 선언  class Point {  int x, y; //한 점 (x,y) 좌표값  public:  void set(int x, int y) { this->x = x; this->y = y; }  void showPoint() {  cout << "(" << x << "," << y << ")" << endl;  }  };  class ColorPoint : public Point { // 2차원 평면에서 컬러점을 표현하는 클래스 ColorPoint.Point를 상속받음  string color;// 점의 색 표현  public:  void setColor(string color) { this->color = color; }  void showColorPoint();  };  void ColorPoint::showColorPoint() {  cout << color << ":";  showPoint(); // Point의 showPoint() 호출  }  int main() {  Point p; // 기본 클래스의 객체 생성  ColorPoint cp; // 파생 클래스의 객체 생성  cp.set(3, 4); // 기본 클래스의 멤버 호출  cp.setColor("Red"); // 파생 클래스의 멤버 호출  cp.showColorPoint(); // 파생 클래스의 멤버 호출  } |

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

**업캐스팅(자식클래스 포인터가 부모클래스 포인터에 치환되는 것)**

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 만화 영화이(가) 표시된 사진

자동 생성된 설명

다운 캐스팅: 부모클래스의 포인터에서 자식클래스의 포인터로 치환

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

<https://rehtorb-algorithm.tistory.com/8>

private과 다르게 파생 클래스에서는 접근 가능하게 해주는 접근 지정자가 protected이다.

텍스트, 도표, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명2,3,4,5,6에서 오류가 나는 이유

3, 4 protected로 접근불가하기 때문에 오류가 발생한다. > p.set(5, 5);

5,6 ColorPoint 클래스에서 cp.x 및 cp.y와 같이 멤버에 직접 액세스하려고 하고 있지만, 이러한 멤버는 Point 클래스에서 protected로 정의되었으므로, protected 멤버는 파생 클래스 내에서 액세스할 수 있지만 클래스 외부에서 직접 액세스할 수는 없다. > cp.set(10, 10);

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

상속의 종류

상속 선언 시 public, private, protected의 3가지 중 하나 지정

◼ 기본 클래스의 멤버의 접근 속성을 어떻게 계승할지 지정

◼ public – 기본 클래스의 protected, public 멤버 속성을 그대로 계승

◼ private – 기본 클래스의 protected, public 멤버를 private으로 계승

◼ protected – 기본 클래스의 protected, public 멤버를 protected로 계승

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

1, 4 오류 > private멤버기 때문에 오류 발생

2 5 번 오류 > protected이므로 오류 발생

3번 오류 > public인데 컴파일 오류 발생 파생클래스에서 객체화 시킬 때 protected로 변환되고 오류가 남.

6번 > 고유 public이라 오류X

다중상속(부모 클래스가 여러 개일경우) – 자바는 x, C++이라 가능

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

|  |
| --- |
| #include <iostream>  using namespace std;  class Adder {  protected:  int add(int a, int b) { return a + b; }  };  class Subtractor {  protected:  int minus(int a, int b) { return a - b; }  };  class Calculator : public Adder, public Subtractor {  public:  int calc(char op, int a, int b);  };  int Calculator::calc(char op, int a, int b) {  int res = 0;  switch (op) {  case '+': res = add(a, b); break;  case '-': res = minus(a, b); break;  }  return res;  }  int main() {  Calculator handCalculator;  cout << "2 + 4 = "  << handCalculator.calc('+', 2, 4) << endl;  cout << "100 - 8 = "  << handCalculator.calc('-', 100, 8) << endl;  } |

가상상속(기본 클래스의 멤버가 중복하여 생성되는 것을 방지)

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명