**메서드 호출**

|  |  |
| --- | --- |
| 1 | (가)~(아)까지 빈 칸에 알맞는 코드를 작성하세요  **public** **class** **Hero** {  **int** hp=10;  **boolean** fly=**false**;  **String** name="메가맨";  **Bullet** bullet;    **public** **void** setHp(가) { //hp 값을 변경하고 싶다  }  **public** **void** setFly(나) {//fly 값을 변경하고 싶다  }  **public** **void** setName(다) {//name 값을 변경하고 싶다  }  **public** **void** fire(라){//bullet 을 다른 무기로 변경하고 싶다  }  **public** **static** **void** main(**String**[] args) {  **Hero** hero = **new** **Hero**();  hero.setHp(마);  hero.setFly(바);  hero.setName(사);  hero.fire(아);  }  } |
| 2 | 다음 설명 중 틀린 것은?  (1)자바의 기본 자료형은 3개이다.  (2)객체 자료형도 자료형 이므로 자바의 모든 자료형의 갯수는 사실상 4개로 보아야 한다  (3)객체 자료형은 메서드의 인수로 전달 되어질 수 없다.  (4)메서드 호출 시 객체 자료형을 인수로 전달하는 방법을 call by reference 라 한다 |
| 4 | 다음 설명 중 틀린 것은?  **class** **Computer** {  **String** color = "yellow";  **int** price = 50;  }  **class** **UseComputer** {  **public** **void** setColor(**Computer** p){ //(가)  p.color="red";  }  **public** **void** setPrice(**int** price) {  price=5; //(나)  }    **public** **static** **void** main(**String**[] args) {  **Computer** com = **new** **Computer**();  com.color = "black";  **UseComputer** up = **new** **UseComputer**();  up.setColor(com); // (다)  up.setPrice(com.price);//(라)    **System**.***out***.println(com.price);// (마)  }  }  (1)(가)의 메서드 인수로 넘겨지는 컴퓨터는 색상이 빨간색으로 변경된다  (2)(라)에서 com 변수가 가리키는 컴퓨터의 값이 (나)로 전달되어 지므로 이 컴퓨터의 가격이 5로  변경 된다  (3)(마)의 출력 결과는 50이다, 즉 컴퓨터의 가격에는 변함이 없다.  (4)(다)는 call by reference 이다  (5)(라)는 call by value 이다  (6) call by reference 에 의해 넘겨진 객체는, 해당 메서드에 의해 영향을 받는다.  (7) call by value 에 의해 넘겨진 값은, 그 값을 보유했던 객체에 영향을 끼친다 |
| 7 | 아래 코드에 대한 설명 중 틀린 것은?  **class** Computer{  **int** speed=300;  }  **class** UseComputer {  **public** **void** setting(Computer c , **int** s){  c.speed=s;  s=50; //(다)  }    **public** **static** **void** main(String[] args) {  **int** k=500; //(가)    UseComputer uc = **new** UseComputer();  Computer com = **new** Computer();//(바)  com.speed=100;    uc.setting(com , k); //(나)    System.*out*.println(com.speed); //(라)  System.*out*.println(k); //(마)    }  }  (1) (가) 변수는 지역변수로서 stack 영역에 쌓인다.  (2) (나)에서 메서드 호출시 com변수에는 객체의 주소값이, k변수에는 변수의 값이 들어있다.  (3) (다)에 의해 (가)의 변수값이 50으로 변경될 것이다.  (4) (라)에서 출력되는 결과는 100 이다.  (5) (마)에서 출력되는 결과는 500 이다. |