**접근 제어**

|  |  |
| --- | --- |
| 1 | 아래와 같은 계좌 클래스에 대한 설명 중 틀린 것은?  **public** **class** **Account** {  **int** balance=200; //(가)  **String** bank="기업은행";  **String** accountNum="12-456-789-10";    **public** **int** getBalance() { //(나)  **return** balance;  }  **public** **void** setBalance(**int** balance) { //(다)  **this**.balance=balance;  }  }  (1) (가)의 balance 를 외부의 어떤 클래스가 사용하려면 .점찍고 접근하면 된다. x  (2) balance를 외부의 어떤 클래스가 참조하려면 (나)의 메서드를 호출하면 잔고값을 얻어갈 수 있다 o  (3) 외부의 클래스가 점찍고 balance 변수에 접근하면 됨에도 불구하고, (나),(라)처럼 값을 반환받거나  변경하기 위한 메서드를 정의하는 것은 시간 낭비이다. x  (4) private 으로 멤버 변수를 제한해두면 외부의 클래스에서 변수에 대해 접근 자체가 불가능해지며,  이렇게 변수 자체를 아예 사용 못하게 하는 것이 자바 보안의 목적이다. x  (5) 메서드를 통해 클래스가 가진 변수를 접근하는 방법은 좋은 방법이다.o |
| 2 | 다음 설명 중 틀린 것은?  (1) 클래스의 멤버 변수는 결국 데이터 이므로 보안상 보호되어야 할 대상이다. o  (2) private 으로 선언된 멤버 변수는 같은 인스턴스내의 멤버 메서드라 할 지라도, 접근이 불가능하다. x  (3) private 으로 멤버 변수를 제한해두면 외부의 클래스에서 변수에 대해 접근 자체가 불가능해지며,  이렇게 변수 자체를 아예 사용 못하게 하는 것이 자바 보안의 목적이다. x  (4) (3)번의 경우처럼 변수 자체에 대한 사용을 금지시킬 경우, 클래스를 정의한 목적 자체가 무의미해  지므로, 변수 사용에 대한 접근 허용을 개발자의 선택에 따라 임의로 조절 가능하도록 하기 위해,  메서드를 통해 간접 사용하게 하는 방식을 객체지향에서는 은닉화 또는 캡슐화(=encapsulation)  라고 한다.o  (5) private으로 접근을 제한한 변수에 대해 값을 변경할 수 있도록 제공되는 메서드를 setter라 하고  값을 접근하여 반환할 수 있도록 제공되는 메서드를 getter 라 한다. o |
| 3 | 아래의 클래스를 은닉화 하세요.  class Customer{  private String name;  private int age;  private boolean isMarry;  public void setName(String name){  this.name = name;  }  public String getName(){  return name;  }  public void setAge(int age){  this.age = age;  }  public int getAge(){  return age;  }  public void setIsMarry(boolean isMarry){  this.isMarry = isMarry;  }  public boolean getIsMarry(){  return isMarry;  }  } |
| 4 | 아래의 클래스를 은닉화 하세요.  class Company{  int memberCount;  String location;  } |