**로고학습일지**

**kt ds University 자바 기반의 데이터 사이언티스트 양성과정**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 학습일시 | 2018. 07. 24 (화) | 장소 | kt ds University B관 201호 | **시 간** | 09:00~18:00 |
| 학습범위 | Java Programming | | | | |
| 작 성 자 | 권석현 | | | **강 사** | 장민창 강사 |

|  |  |
| --- | --- |
| 학습안건 | 메소드, 클래스, 생성자 |

|  |  |
| --- | --- |
| 학습내용 | 내용 |
| 접근제한자.  public  protected (잘안씀)  default  private  접근제한자는 클래스, 멤버변수, 메소드에 사용할 수 있다.  멤버변수 선언 -> private (많이사용)  메소드 -> public (많이사용)  private선언하면 클래스 안에서만 가능  다른 클래스에서 객체로 만들어서 못씀  private int a;  public void setA(int a){  a = a;  } //이렇게 쓴다 private로 선언되서 사용못하니 퍼블릭 메소드 안에 넣어서 쓴다.  //저렇게 쓰면 private int a에 들어갈 값을 필터링 해 줄수도 있다. (encapsulation)  public void setHealthPoint( int healthPoint ) { // setter    if ( healthPoint <= 10000 ) {  //this ==> Member (클래스가 가지는멤버 : 멤버변수, 메소드)  //this ==> Instance (객체)  this.healthPoint = healthPoint;  멤버변수는 항상 this. 붙이면 가독성이 높아짐, 헷갈리지도 않는다.  멤버변수랑 아규먼트값 같으면 아규먼트로 인식함. this 붙여야됨  // 알트 시프트 에스 제너레이트 게터엔세터 하면 게터세터메서드 체크해서 자동으로 만들수 있다.  컨트롤 + 쉬프트 + f 자동 줄 정리  멤버변수 하나에 게터, 세터 하나씩 필요함.  리스트 에 리스트를 그냥 넣으면  메모리주소가 같다. 즉 한 리스트의 값이 바뀌면 두개 다 바뀜.  얕은복사, 깊은복사가 있다.  public void setHealthPotionList(List<String> healthPotionList) {  this.healthPotionList = new ArrayList<String>();  this.healthPotionList.addAll(healthPotionList);  //이런식으로 해야됨. addAll() 메소드는 괄호안에있는 리스트의 모든 값들을 가져온다 하나씩 포문돌린것처럼  public List<String> getHealthPotionList(){  List<String> healthPotionListCopy = new ArrayList<String>();  healthPotionListCopy.addAll(this.healthPotionList);  return healthPotionListCopy;  //gettet메소드는 이런식으로 그냥저장하면 메모리주소 곂쳐서 같은주소를 공유하니까  private의 변수는 항상 게터세터가필요. 자기 클래스에서밖에 접근을 못하니까  wrapper클래스와 String은 메모리를 가지지 않는다. 예외적으로 값을 가짐.  생성자  생성자는 멤버변수를 초기화 할 수 있게 해준다.  ex) public Calculator();{ //클래스명과 동일하게 해야함.  a = 10;  b = 30;  생성자는 보통 멤버변수들 아래에 만든다  초기값들.  리스트는 객체생성 안해서 null이다.  private int number; // 0  private boolean inHuman; // false  private double floatingNumber; // 0.0  private long longNumber; //0L  private float floatingNumber2; //0.0F  private char character; // ''  private String name; //null  private String cpu; // null  private List<String> ramList; //null  private List<String> hddList; //null  private List<String> ssdList; //null  레퍼런스 타입들은 초기화가 필요하다.  Constructor Overloading  생성자 오버로딩  public GameObject(int healthPoint, int manaPoint, int damagePoint) {  같은 이름을 가진 메소드이지만 아귀먼츠가 다를 경우.  this(); 이렇게 쓰면 디폴트 생성자 실행하라는 의미.  생성자 오버로딩 사용시 중복된 코드를 줄일 수 있는 장점. |

|  |  |
| --- | --- |
| 특이사항 |  |