注意事项和细节讨论

1) 多态的前提是：两个对象(类)存在继承关系

2) 多态的向上转型：

①本质：父类的引用指向了子类的对象

②语法：父类类型 引用名 = new 子类类型();

③特点：编译类型看左边，运行类型看右边

可以调用父类中的所有成员(需遵守访问权限)；

不能调用子类中特有成员；

最终运行效果看子类的具体实现！

3) 多态的向下转型：

①语法：子类类型 引用名 = (子类类型)父类引用；

②只能强转父类的引用，不能强转父类的对象

③要求父类的引用必须指向的是当前目标类型的对象

④可以调用子类类型中的所有成员

4) 属性没有重写之说！属性的值看编译类型 (此处与方法的原则不同)

5) instanceOf 比较操作符，用于判断对象的类型是否为 XX类型 或 XX类型的子类型

此处指运行类型

```
请说出下面的每条语言，哪些是正确的，哪些是错误的，为什么？
public class PolyExercise01{
        public static void main(String[] args) {
                double d = 13.4; ✓
                long l = (long)d; ✓
                System.out.println(l); // 13
                int in = 5; ✓
                boolean b = (boolean)in; ✗
                Object obj = "Hello"; ✓ 向上转型
                String objStr = (String)obj; ✓ 向下
                System.out.println(objStr); // hello

                Object objPri = new Integer(5); ✓ 向上
                String str = (String)objPri; ✗Integer → String → ClassCastException 错误
                Integer str1 = (Integer)objPri; ✓
        }}
```

Java的动态绑定机制 !!!

Java重要特性：动态绑定机制 DynamicBinding.java com.hspedu.poly_.dynamic_

```
class A {//父类
    public int i = 10;
    public int sum() {
        return getI() + 10;
    }
    public int sum1() {
        return i + 10;
    }
    public int getI() {
        return i;
    }
}
```

```
class B extends A {//子类
    public int i = 20;
    public int sum() {
        return i + 20;
    }
    public int getI() {
        return i;
    }
    public int sum1() {
        return i + 10;
    }
}
```

```
//main方法中
A a = new B();//向上转型
System.out.println(a.sum());//?40
System.out.println(a.sum1());//30
2min
```

java的动态绑定机制

1. 当调用对象方法的时候，该方法会和该对象的内存地址/运行类型绑定
2. 当调用对象属性时，没有动态绑定机制，哪里声明，那里使用

多态的应用

1) 多态数组

数组的定义类型为父类类型，里面保存的实际元素类型为子类类型

2) 多态参数

方法定义的形参类型为父类类型，实参类型允许为子类类型.

P319 - P Java _ Object类

equals 方法

经典面试题： == 与 equals 对比

1) "==" 是一个比较运算符，既可以判断基本类型，又可以判断引用类型

判断基本类型：判断值是否相等 [double a=10.0; int b=10; a==b? // true].

判断引用类型：判断地址是否相等，即判断是否为同一对象

2) equals 是 Object 类中的方法，只能判断引用类型

默认判断的是地址是否相等，子类中往往重写该方法用于判断内容是否相等.

Object 类对象调用 equals 方法是比较地址.

↳ 子类调用是比较内容，例 String 类中重写了 equals 方法

练习： $out("hello" == new java.sql.Date()); // 编译错误

hashCode 方法

1) 提高具有哈希结构的容器的效率

2) 两个引用，如果指向的是同一个对象，则哈希值肯定是一样的

3) 哈希值主要根据地址号来的，但不能完全将哈希值等价于地址.

toString 方法

1) 默认返回：全类名 + @ + 哈希值的十六进制

2) 子类往往重写 toString 方法，用于返回对象的属性信息

3) 重写 toString 方法，打印对象或拼接对象时，都会自动调用该对象 toString 形式

4) 当直接输出一个对象时，toString 方法会被默认的调用.

finalize 方法

当垃圾回收器确定不存在对该对象的更多引用时，由对象的垃圾回收器调用此方法

1) 当对象被回收时，系统自动调用该对象的finalize方法。子类可以重写该方法，做一些释放资源的操作.

2) 什么时候被回收：当某个对象没有任何引用时，则jvm就认为这个对象是一个垃圾对象，就会使用垃圾回收机制来销毁对象，在销毁对象前，会先调用finalize方法

3) 垃圾回收机制的调用，由系统来决定，也可以通过System.gc()主动触发垃圾回收机制.