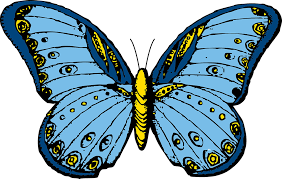
The Java OCA Exam covers the basics, with twists. Let's look at Inheritance and Overriding.



**class** Parent {

**public** **int** theVar = 9;

**public** **void** methodA(){

System.***out***.println("Method A in Parent");

}

}

**class** Child **extends** Parent {

**public** **int** theVar = 99;

**public** **void** methodA(){

System.***out***.println("Method A in Child");

}

}

**class** Test {

**public** **static** **void** main (String args[]){

Parent p = **new** Child();

p.methodA();

System.***out***.println("Calling var "+p.theVar);

}

}

**What happens when we run this?**

We print out "Method A in Child" and the next line prints out 9. The Parent reference p accessed the parent class for the instance variable and the child class for the instance method.

In Java, non-static methods are based on the run-time type of the object. So, since p references the Child object we'll call the method in the Child class. This rule does not hold up for instance variables, so the p reference for instance variable theVar will access it from the parent class even if the p references a Child object.

**What if a method exists in the parent class but not the child class?**

Remember, the rule is that since p references the Child object, we'll call methods from the Child class. In this case, there is no methodB in the child class. Since the p reference does not find methodB in the child class, it will move up the inheritance chain to find methodB in the parent class. The output will include "Method B in Parent".

**class** Parent{

**public** **int** parentVar = 9;

**public** **void** methodA(){

System.***out***.println("Method A in Parent");

}

**public** **void** methodB(){

System.***out***.println("Method B in Parent");

}

}

**class** Child **extends** Parent{

**public** **int** parentVar = 99;

**public** **void** methodA(){

System.***out***.println("Method A in Child");

}

}

**class** Test{

**public** **static** **void** main(String args[]){

Parent p = **new** Child();

p.methodA();

System.***out***.println("Calling var "+ p.parentVar);

p.methodB();

}

}