```
    this.e = exp;

    this.val = 1;
    if(exp==0) return;
    for( ; exp>0; exp--) this.val = this.val * base;
  }

  double get_pwr() {
    return this.val;
  }
}
```

Actually, no Java programmer would write **Pwr** as just shown because nothing is gained, and the standard form is easier. However, **this** has some important uses. For example, the Java syntax permits the name of a parameter or a local variable to be the same as the name of an instance variable. When this happens, the local name *hides* the instance variable. You can gain access to the hidden instance variable by referring to it through **this**. For example, although not recommended style, the following is a syntactically valid way to write the **Pwr( )** constructor.

```
Pwr(double b, int e) {
  this.b = b;
  this.e = e;

  val = 1;
  if(e==0) return;
  for( ; e>0; e--) val = val * b;
}
```

This refers to the **b** instance variable, not the parameter.

In this version, the names of the parameters are the same as the names of the instance variables, thus hiding them. However, **this** is used to "uncover" the instance variables.

Module 4 Mastery Check

1. What is the difference between a class and an object?
2. How is a class defined?
3. What does each object have its own copy of?
4. Using two separate statements, show how to declare an object called **counter** of a class called **MyCounter**.

5. Show how a method called **myMeth( )** is declared if it has a return type of **double** and has two **int** parameters called **a** and **b**.
6. How must a method return if it returns a value?
7. What name does a constructor have?
8. What does **new** do?
9. What is garbage collection, and how does it work? What is **finalize( )**?
10. What is **this**?
11. Can a constructor have one or more parameters?
12. If a method returns no value, what must its return type be?