Java Lecture-5

Classes and Objects

- A **class** defines a new data type.
- Once defined, this new type can be used to create **objects** of that type.
- Thus, a class is a *template* for an object, and an object is an *instance* of a class.
- When you define a class, you declare its exact form and nature.
- You do this by specifying the data that it contains and the code that operates on that data.
- While very simple classes may contain only code or only data, most real-world classes contain both.

class *classname* {
type instance-variable1;

type instance-variable2; // ...
type instance-variableN;

type methodname1(*parameter-list*) { // body of method

}
type methodname2(*parameter-list*) {

// body of method }

// ...
type methodnameN(*parameter-list*) {

// body of method }

}

```
/* A program that uses the Box class.

   Call this file BoxDemo.java
*/
class Box {
  double width;
  double height;
  double depth;
}

// This class declares an object of type Box.
class BoxDemo {
  public static void main(String args[]) {
    Box mybox = new Box();
    double vol;

    // assign values to mybox's instance variables
    mybox.width = 10;
    mybox.height = 20;
    mybox.depth = 15;

    // compute volume of box
    vol = mybox.width * mybox.height * mybox.depth;

    System.out.println("Volume is " + vol);
  }
}
```

Declaring Objects

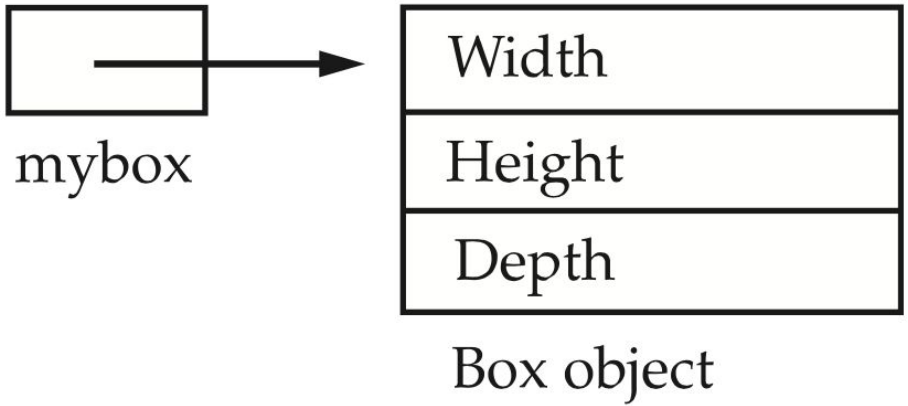
Box mybox = new Box();

This statement combines the two steps just described. It can be rewritten like this to show each step more clearly:

```
Box mybox;                // declare reference to object
mybox = new Box();        // allocate a Box object
```

class-var = new *classname* ( );

the **new** operator dynamically allocates memory for an object.

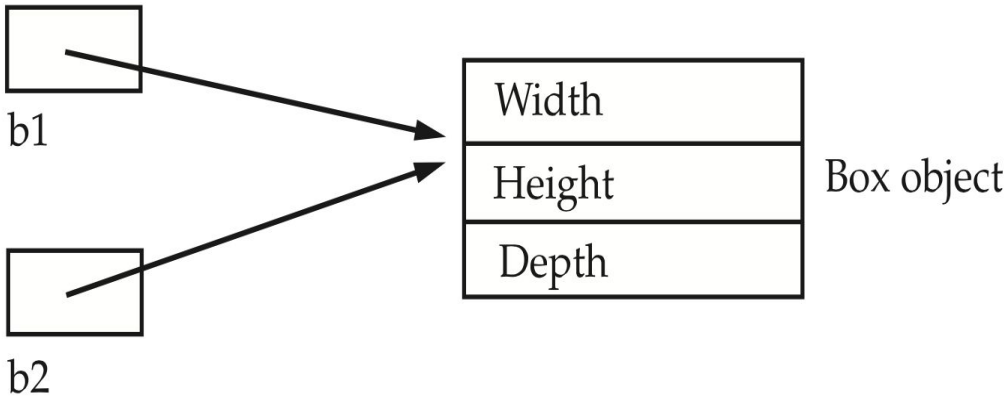
Statement	Effect
Box mybox;	
mybox = new Box();	

Assigning Object Reference Variables

Box b1 = new Box();

Box b2 = b1;

It simply makes **b2** refer to the same object as does **b1**.



Adding a Method to the Box Class

```
// This program includes a method inside the box class.

class Box {
  double width;
  double height;
  double depth;

  // display volume of a box
  void volume() {
    System.out.print("Volume is ");
    System.out.println(width * height * depth);
  }
}

class BoxDemo3 {
  public static void main(String args[]) {
```

```
    Box mybox1 = new Box();
    Box mybox2 = new Box();

    // assign values to mybox1's instance variables
    mybox1.width = 10;
    mybox1.height = 20;
    mybox1.depth = 15;

    /* assign different values to mybox2's
       instance variables */
    mybox2.width = 3;
    mybox2.height = 6;
    mybox2.depth = 9;

    // display volume of first box
    mybox1.volume();

    // display volume of second box
    mybox2.volume();
  }
}
```

This program generates the following output, which is the same as the previous version.

```
Volume is 3000.0
Volume is 162.0
```

Returning a Value

```
class Box {
  double width;
  double height;
  double depth;

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}

class BoxDemo4 {
  public static void main(String args[]) {
    Box mybox1 = new Box();
    Box mybox2 = new Box();
    double vol;

    // assign values to mybox1's instance variables
    mybox1.width = 10;
    mybox1.height = 20;
    mybox1.depth = 15;
```

```
    /* assign different values to mybox2's
       instance variables */
    mybox2.width = 3;
    mybox2.height = 6;
    mybox2.depth = 9;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }
}
```

Adding a Method That Takes Parameters

```
// This program uses a parameterized method.

class Box {
  double width;
  double height;
  double depth;

  // compute and return volume
  double volume() {
    return width * height * depth;
  }

  // sets dimensions of box
  void setDim(double w, double h, double d) {
    width = w;
    height = h;
    depth = d;
  }
}

class BoxDemo5 {
```

```
  public static void main(String args[]) {
    Box mybox1 = new Box();
    Box mybox2 = new Box();
    double vol;

    // initialize each box
    mybox1.setDim(10, 20, 15);
    mybox2.setDim(3, 6, 9);

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }
}
```

Constructors

```
/* Here, Box uses a constructor to initialize the
   dimensions of a box.
*/
class Box {
  double width;
  double height;
  double depth;

  // This is the constructor for Box.
  Box() {
    System.out.println("Constructing Box");
    width = 10;
    height = 10;
    depth = 10;
  }

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}
```

```
class BoxDemo6 {
  public static void main(String args[]) {
    // declare, allocate, and initialize Box objects
    Box mybox1 = new Box();
    Box mybox2 = new Box();

    double vol;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }
}
```

When this program is run, it generates the following results:

```
Constructing Box
Constructing Box
Volume is 1000.0
Volume is 1000.0
```

Parameterized Constructors

```
/* Here, Box uses a parameterized constructor to
   initialize the dimensions of a box.
*/
class Box {
  double width;
  double height;
  double depth;

  // This is the constructor for Box.
  Box(double w, double h, double d) {
    width = w;
    height = h;
    depth = d;
  }

  // compute and return volume
  double volume() {
    return width * height * depth;
  }
}
```

```
class BoxDemo7 {
  public static void main(String args[]) {
    // declare, allocate, and initialize Box objects
    Box mybox1 = new Box(10, 20, 15);
    Box mybox2 = new Box(3, 6, 9);

    double vol;

    // get volume of first box
    vol = mybox1.volume();
    System.out.println("Volume is " + vol);

    // get volume of second box
    vol = mybox2.volume();
    System.out.println("Volume is " + vol);
  }
}
```

The output from this program is shown here:

```
Volume is 3000.0
Volume is 162.0
```

Arrays

- An *array* is a group of like-typed variables that are referred to by a common name.
- Arrays of any type can be created and may have one or more dimensions.
- A specific element in an array is accessed by its index.
- Arrays offer a convenient means of grouping related information.

type array-var = new *type* [*size*];

Int month_days = new int[12];

month_days[1] = 28;

class **AutoArray** {
 public static void main(String args[]) {
  int month_days[] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
  System.out.println("April has " + month_days[3] + " days.");
 }}

```
// Average an array of values.
class Average {
 public static void main(String args[]) {
    double nums[] = {10.1, 11.2, 12.3, 13.4, 14.5};
    double result = 0;
    int i;
for(i=0; i<5; i++)
      result = result + nums[i];
   System.out.println("Average is " + result / 5);
  }
}
```

Multidimensional Arrays

```
class TwoDArray {
 public static void main(String args[]) {
    int twoD[][]= new int[4][5];
    int i, j, k = 0;
   for(i=0; i<4; i++)
      for(j=0; j<5; j++) {
       twoD[i][j] = k;
k++; }
   for(i=0; i<4; i++) {
      for(j=0; j<5; j++)
       System.out.print(twoD[i][j] + " ");
      System.out.println();
} }}
```