Return Values in a Method

If you want the method to return a value, you can use a primitive data type (such as int or double) instead of void, and use the return keyword inside the method:

Example

```
public static void main(String[] args)
{
   int total = calculate(200, 300);

   System.out.println();
   System.out.println("The total amount = " + total);
}

public static int calculate(int a, int b)
{
   return a + b;
}
```

Example

```
public static String getName(String firstName, String lastName)
{
   String name = firstName + " " + lastName;
   return name;
}

public static void main(String[] args)
{
   String name1 = getName("John", "Mayer");
   String name2 = getName("Mary", "Hopkin");

   System.out.println(name1 + " is a singer of United States.");
   System.out.println(name2 + " is a singer of United Kingdom.");
}
```

Method Overloading

Generally, we can declare different methods as the following statements to do some things:

```
int myMethod1(int x, int y)

float myMethod2(float x, float y)

double myMethod3(double x, double y)
```

Consider the following example, which have two methods that add numbers of different type:

Example

```
public static int myMethod1(int x, int y)
{
   return x + y;
}

public static double myMethod3(double x, double y)
{
   return x + y;
}

public static void main(String[] args)
{
   int iNumber = myMethod1(20, 50);
   double dNumber = myMethod3(3.725, 6.148);

   System.out.println("iNmuber: " + iNumber);
   System.out.println("dNumber: " + dNumber);
}
```

Instead of defining two methods that should do the same thing, it is better to overload one. With method overloading, multiple methods can have the same name with different parameters:

Example

```
public static int myMethod(int x, int y)
{
   return x + y;
}

public static double myMethod(double x, double y)
{
   return x + y;
}

public static void Main(String[] args)
{
   int iNumber = myMethod(20, 50);
   double dNumber = myMethod(3.725, 6.148);

   System.out.println("iNmuber: " + iNumber);
   System.out.println("dNumber: " + dNumber);
}
```