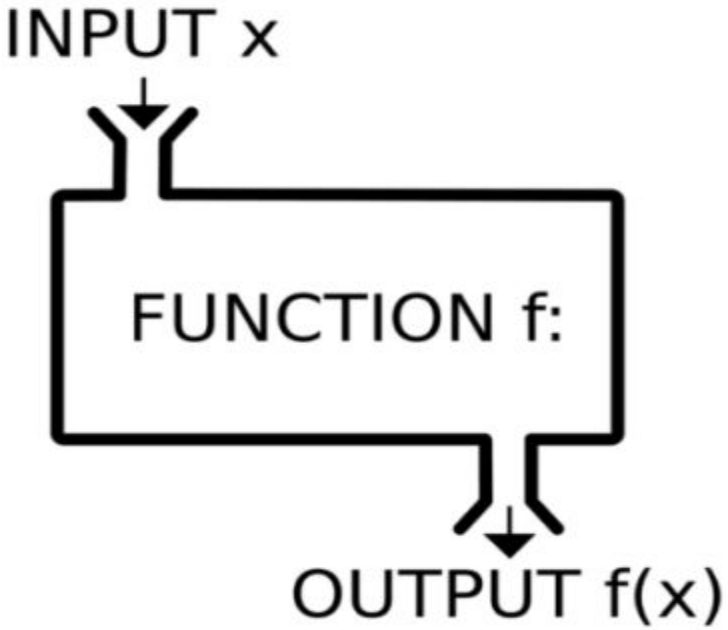
Lecture 10

Java Methods/ Arrays

Today’s Topics

- What are functions?
- How to call functions?
- Variable Scope
- Pass-by-value and Pass-by-reference
- Function Call Stack
- Arrays

What's a function?



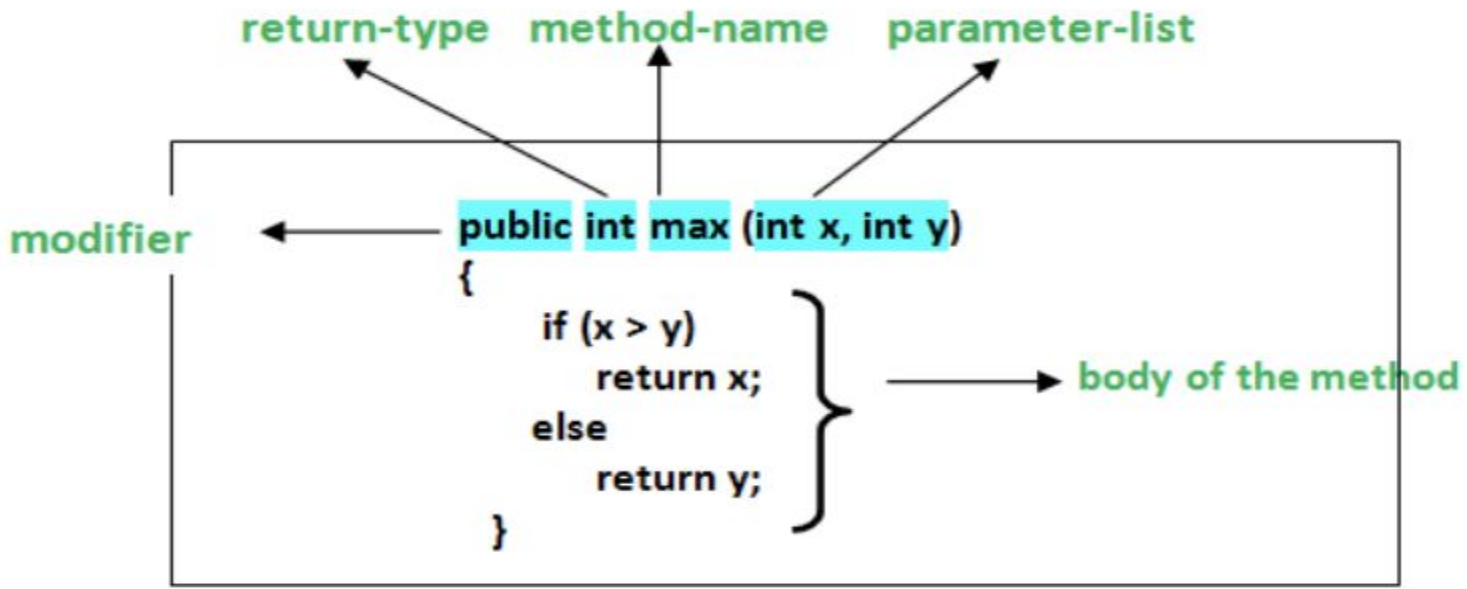
Why do we need functions?

Functions vs Methods

- **Methods:** Is associated to the instance of the object it is called using.
- **Functions**: Called independently by their name.

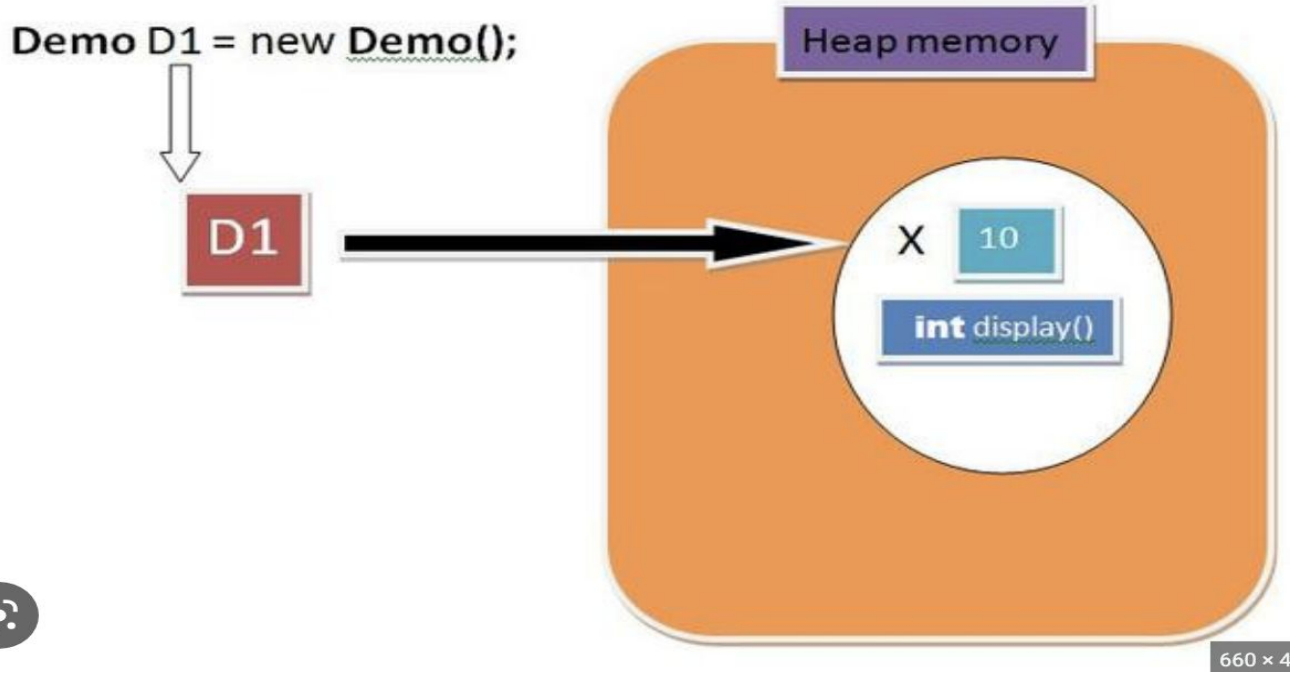
```java
class Main {
    public static void main(String[] args) {
        Car ishansCar = new Car();
        ishansCar.startCar();
        ishansCar.stopCar();

        Car namansCar = new Car();
        namansCar.startCar();
        namansCar.moveForward();
        namansCar.moveRight();
        namansCar.stopCar();

        Car dineshsCar = new Car();
        dineshsCar.startCar();
        dineshsCar.moveLeft();
        dineshsCar.stopCar();


        Car.hasAirBags();
    }
}

// Non-static Methods are associated with an object
// static methods are associated with a class

class Car {
    // data
    String color;
    int tyres;
    int windows;
    int seats;
    int gears;

    // methods

    public static void hasAirBags() {
        //puts emergency brake
    }

    // non-static methods -- associated with an object
    public void startCar() {
        // starts the engine
    }

    public void stopCar() {
        // stops the engine
    }

    public void moveForward() {
        // moves forward
    }

    public void moveLeft() {
        // moves left
    }

    public void moveRight() {
        // moves right
    }

    public void moveBackward() {
        // moves backward
    }
}
```

Method Definition



```
class Main {
   public static void main(String[] args) {
        Duck manny = new Duck(); // what will be the size of manny object?
   }
}

class Duck {
   int weight;  // 4 bytes
   int height;  // 4 bytes
   int color;   // 4 bytes
}
```

What's a reference?



```java
class Main {
   // Primitive DataTypes (int, float, char) - Store value
   // Non-Primitive (String, Duck) - Store the address (for the object) which will store values
   public static void main(String[] args) {
       int age = 33; // age will store the value 33
       float salary = 14.00 // salary will store the value 14.00

       String name = "Ishan"; // name will store the address of memory which will contain the value "Ishan"
       // manny will store the address of new duck object.
       Duck manny = new Duck(); // what will be the size of manny object?

       // When a variable stores an address we call that variable 'reference'
       /** new Duck();
        1) This allocates storage for a new object of type Duck
        2) This storage has an address also
       **/
   }
}

class Duck {
   int weight;  // 4 bytes
   int height;  // 4 bytes
   int color;   // 4 bytes
}
```

pass-by-value vs pass-by-reference

```java
class Main {
   // reference variables = duck
   // non-reference variables = salary

   public static void main(String[] args) {
       double salary =  14.00; //  salary will store the value 14.00
       System.out.println( "Salary before calling updateSalary = "    + salary);
       updateSalary(salary);   // value 14.00 is passed
       System.out.println( "Salary after calling updateSalary = "    + salary);

       // ------------------------------

       Duck duck =  new Duck();  // duck will have the address for the new duck object
       duck.height =  14;
       System.out.println( "Height of duck before calling updateDuck = "    + duck.height);
       updateDuck(duck);   // address of duck object is passed to the function
       System.out.println( "Height of duck after calling updateDuck = "    + duck.height);
   }

   public static void updateSalary( double salary) {
       salary =  25.00;
   }

   public static void updateDuck(Duck duck) {
       duck.height =  7;
   }

   public static class Duck {
       int height;
       int weight;
   }
}
```

```java
class Main {
   // reference variables = duck
   // non-reference variables = salary

   public static void main(String[] args) {
       double salary = 14.00; //  salary will store the value 14.00
       // salary = 14.00
       System.out.println("Salary before calling updateSalary = " salary);
       updateSalary(salary);// value 14.00 is passed
       System.out.println("Salary after calling updateSalary = " salary);

       // ------------------------------


       Duck duck =new Duck(); // duck will have the address for the new duck object
       // duck = 0x123456 - this address will have the actual object

       System.out.println("Address is "+ duck );
       duck.height =14;
       System.out.println("Height of duck before calling updateDuck =+"duck.height);
       updateDuck(duck);// address of duck object is passed to the function - 0x123456
       System.out.println("Height of duck after calling updateDuck =+"duck.height);
   }

   public static void updateSalary(double salary) {
       salary =25.00;
   }

   public static void updateDuck(Duck duck) {
       duck.height =7;
   }

   public static class Duck {
       int height;
       int weight;
   }
}
```

```java
class Main {
   // reference variables = name
   // non-reference variables = age, salary
   public static void main() {
       String name ="Ishan Sharma"; //value for name is the address of the location where String Ishan Sharma is stored.
       int age = 33; //age will store the value 33
       float salary =14.00 //  salary will store the value 14.00

       printName(name);// The address stored in name variable is passed as a value.
       printAge(age);// Value 33 is passed to printAge
       printSalary(salary);// Value 14.00 is passed to printSalary
   }

   // pass-by-reference
   public static void printName(String name) {
       System.out.println("Name is " + name);
   }

   // pass-by-value
   public static void printAge(int age) {
       System.out.println("Age is " + age);
   }

   // pass-by-value
   public static void printSalary(float salary) {
       System.out.println("Salary is "+ salary);
   }
}
```