Object Oriented Programming using Java

Prepared By:
Suyel, PhD
Assistant Professor
Dept. of CSE, NIT Patna

Outline

1. Variable

2. Examples of Operators

Variable

- A variable can be considered as a name given to the location in memory where values are stored.
- A variable is assigned with a data type.
- In Java, all the variables must be declared before use.
- The value stored in a variable can be changed during program execution.
- Basic Syntax:

<datatype> <variable_name>

Variable (Cont…)

- **Variable Name (Cont…)**
  - **Can you answer that whether the below lines are correct or not?**
    - int shrift = 0;
    - char thisMustBeTooLong;
    - int bubble = 0, toil = 9, trouble = 8
    - int 8ball;
    - int double;

Variable (Cont…)

- **Variable Name (Cont…)**
  - **Answers of the previous slide:**
    - int shrift = 0; //OK
    - char thisMustBeTooLong; //OK in syntax, but poor
    - int bubble = 0, toil = 9, trouble = 8 // “;” missing at the end
    - int 8ball; //Can’t start with a digit
    - int double; //double is a reserve word

Variable (Cont…)

- There can be three types of variables in a class:
  - **Local variable:** Local variables are declared within the method. These variables are destroyed, when method is completed. A local variable cannot be defined with “static” keyword.
  - **Instance variable:** Instance variables are declared within a class, but outside any method, constructor or block. These type of variables can be accessed from inside any method.
  - **Static variable:** Static variables are declared within a class with the “static” keyword, but outside any method. If changes are made to this static variable, all other instances can see the effect of the changes. Memory allocation for static variable happens only once, when the class is loaded in the memory.

Variable (Cont…)

- **Example of variables**

```
class Hello
{
    int x = 10;                    //Instance variable
    static int y = 20;             //Static variable
    void hi()
   {
       int z = 30;                 //Local variable
   }
}
```

Variable (Cont…)

- **Example of local variable**

```
class LocalVariable
{
    void CountAge()
    {
        int age = 18;       //Local variable
        age = age + 5;
        System.out.println("Student age is: " + age);
    }

    public static void main(String args[])
    {
        LocalVariable obj = new LocalVariable();
        obj.CountAge();
    }
}
```

Variable (Cont…)

- **Example of local variable (Cont…)**

  - **Output**

    Student age is: 23

Variable (Cont…)

- **Example of instance variable**

```
class Counter
{
      int count=0;          //instance variable
      void Counter1()
     {
           count++;
           System.out.println(count);
     }
     public static void main(String args[])
    {
         Counter C1=new Counter();
         Counter C2=new Counter();
         Counter C3=new Counter();
         C1.Counter1();
         C2.Counter1();
         C3.Counter1();
    }
}
```

Variable (Cont…)

- **Example of instance variable (Cont…)**

  - **Output**

    1

    1

    1

Variable (Cont…)

- **Example of static variable**

```
class CounterStatic
{
   static int count=0;          //Will get memory once and retain its value
   void Counter1()
   {
      count++;                  //Incrementing the value of static variable
      System.out.println(count);
   }
   public static void main(String args[])
   {
      CounterStatic C1=new CounterStatic();
      CounterStatic C2=new CounterStatic();
      CounterStatic C3=new CounterStatic();
      C1.Counter1();
      C2.Counter1();
      C3.Counter1();
   }
}
```

Variable (Cont…)

- **Example of static variable (Cont…)**

  - **Output**

    1

    2

    3

Variable (Cont…)

Combination of Variables

Student.java

```
class Student
{
    int rollno;                          //Instance variable
    String name;
    static String college = “ITS”;       //Static variable
    void  Student1(int r, String n)
    {
        rollno = r;
        name = n;
    }
    void display ()
    {
         System.out.println(rollno+" "+name+" "+college);
    }
}
```

Test.java

```
public class Test
{
    public static void main(String args[])
    {
        Student s1 = new Student();
        Student s2 = new Student();
        s1.Student1(111, “Karan”);
        s2.Student1(222, “Aryan”);
        s1.display();
        s2.display();
    }
}
```

Variable (Cont…)

- **Combination of Variables (Cont…)**

  - **Output**

    111 Karan ITS
    222 Aryan ITS

Variable (Cont…)

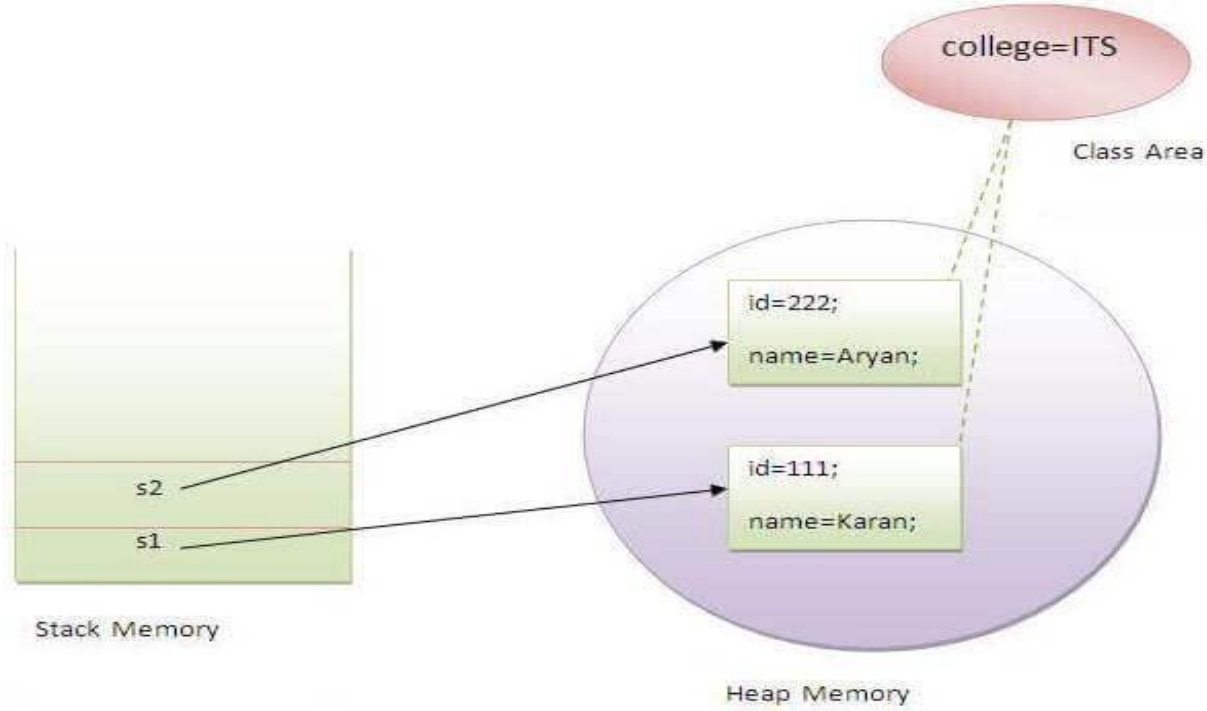

Fig. 1: Memory representation

Examples of Operators

```
class Operatorexample
{
   public static void main(String args[])
   {
      int x=100;
      System.out.println(x++);
      System.out.println(++x);
      System.out.println(x--);
      x=200;
      System.out.println(--x);
   }
}
```

Examples of Operators (Cont…)

- **Output**

  100

  102

  102

  199

Examples of Operators (Cont…)

```
class Operatorexample1
{
   public static void main(String args[])
   {
      System.out.println(10*9/5+2-1*8/2);
   }
}
```

Examples of Operators (Cont…)

- **Output**

  16

Examples of Operators (Cont…)

```
class Operatorexample2
{
   public static void main(String args[])
   {
      System.out.println(10<<2);
      System.out.println(-10<<2);
      System.out.println(20>>2);
      System.out.println(20>>>2);
      System.out.println(-20>>2);
      System.out.println(-20>>>2);
   }
}
```

Examples of Operators (Cont…)

- **Output**

```
 40
-40
 5
 5
-5
 1073741819
```

Examples of Operators (Cont…)

- **Working of “>>” and “>>>”**
  - For unsigned both give the same result.
  - For signed “>>”, compiler stores the sign bit (left most bit) and adds the left most bits with 1.
  - For signed “>>>”, compiler does not store the sign bit (left most bit) and adds the left most bits with 0.

Examples of Operators (Cont…)

Working of “>>” and “>>>” (Cont…)

- To understand the process, we need to know the representation of a negative number in memory.
- Let the number is -10. -10 is stored in the memory in its 2’s complement form. There are the following steps to get 2’s complement:

1. Convert the number into its binary form.

   1010

2. Convert the binary form of the given number into 1’s complement.

   1111 1111 1111 1111 1111 1111 1111 0101 //32-bit representation

3. Add 1 to the resultant of the above step.

   1111 1111 1111 1111 1111 1111 1111 0110 //-10 is stored in the memory in this form

Examples of Operators (Cont…)

- **Working of “>>” and “>>>” (Cont…)**
  - If -10>>2, then compiler would calculate the 2’s complement of memory representation of -10.

    1111 1111 1111 1111 1111 1111 1111 0110 //Memory representation of -10

    1111 1111 1111 1111 1111 1111 1111 1101 //2 bit right shift and add the left most bits with 1.

    0000 0000 0000 0000 0000 0000 0000 0010 //1’s complement

    0000 0000 0000 0000 0000 0000 0000 0011 //2’s complement (1 is added)

    As the compiler stores the sign bit, the answer would be -3.

Examples of Operators (Cont…)

- **Working of “>>” and “>>>” (Cont…)**
  - If -10>>>2:

1111 1111 1111 1111 1111 1111 1111 0110 //Memory representation of -10

0011 1111 1111 1111 1111 1111 1111 1101 //2 bit right shift and adds the left most bits with 0.

As the answer is positive after 2 bit right shift, complier would not calculate the 2’s complement. Thus, the answer would be 1073741821. Here, compiler does stores the sign bit.

Examples of Operators (Cont…)

```
public class Operatorexample7
{
   public static void main(String args[])
   {
      int a = 10;
      int b = -10;
      System.out.println(Integer.toBinaryString(a));
      System.out.println(Integer.toBinaryString(~a));
      System.out.println(Integer.toBinaryString(~a+1));
      System.out.println(Integer.toBinaryString(b));
   }
}
```

Examples of Operators (Cont…)

- **Output**

Examples of Operators (Cont…)

```
public class Operatorexample8
{
   public static void main(String args[])
   {
      int a = 10;
      int b = -10;
      System.out.println(Integer.toBinaryString(a));
      System.out.println(Integer.toBinaryString(a>>>2));
      System.out.println(b);
      System.out.println(Integer.toBinaryString(b));
      System.out.println(Integer.toBinaryString(b = b>>>2));
      System.out.println(b);
   }
}
```

Examples of Operators (Cont…)

- **Output**

Examples of Operators (Cont…)

```
class Operatorexample3
{
   public static void main(String args[])
   {
      int a=10;
      int b=5;
      int c=20;
      System.out.println(a<b&&a<c);
      System.out.println(a<b&a<c);
      System.out.println(a<b||a<c);
      System.out.println(a<b|a<c);
      System.out.println(a>b||a++<c);
      System.out.println(a);
      System.out.println(a>b|a++<c);
      System.out.println(a);
   }
}
```

Examples of Operators (Cont…)

- **Output**

```
false
false
true
true
true
10
true
11
```

Examples of Operators (Cont…)

```
class Operatorexample4
{
   public static void main(String args[])
   {
        int a=3;
        int b=6;
        int c=(a<b)?a:b;
        System.out.println(c);
   }
}
```

Examples of Operators (Cont…)

- **Output**

  3

Examples of Operators (Cont…)

```
class Operatorexample5
{
   public static void main(String args[])
   {

        int a=12;
        a+=4;
        System.out.println(a);
        a-=3;
        System.out.println(a);
        a*=4;
        System.out.println(a);
        a/=5;
        System.out.println(a);
   }
}
```

Examples of Operators (Cont…)

- **Output**

16

13

52

10

Examples of Operators (Cont…)

```
class Operatorexample6
{
   public static void main(String args[])
   {

        int a=10;
        int b=20;
        System.out.println("a == b =" + (a == b));
        System.out.println("a != b =" + (a != b));
        System.out.println("a > b =" + (a > b));
        System.out.println("a < b =" + (a < b));
        System.out.println("a >= b =" + (a >= b));
        System.out.println("a <= b =" + (a <= b));
   }
}
```

Examples of Operators (Cont…)

- **Output**

  a == b = false

  a != b =true

  a > b =false

  a < b =true

  a >= b =false

  a <= b =true


thank
you . . .

Slides are prepared from various sources, such as Book, Internet Links and many more.