15. Recursion

Recursion in java is a process in which a method calls itself continuously. A method in java that calls itself is called recursive method.

It makes the code compact but complex to understand.

Syntax:

```
returntype  methodname(){
//code to be executed
methodname();//calling same method
}
```

Java Recursion Example 1: Infinite times

```
public class RecursionExample1 {
static void p(){
System.out.println("hello");
p();
}

public static void main(String[] args) {
p();
}
}
```

Output:

```
hello
hello
...
java.lang.StackOverflowError
```

Java Recursion Example 2: Finite times

```
public class RecursionExample2 {
static int count=0;
static void p(){
count++;
if(count<=5){
System.out.println("hello "+count);
p();
}
}
```

```
10. public static void main(String[] args) {
11. p();
12. }
13. }
```

Output:

```
hello 1
hello 2
hello 3
hello 4
hello 5
```

Java Recursion Example 3: Factorial Number

```
1.  public class RecursionExample3 {
2.     static int factorial(int n){
3.           if (n == 1)
4.             return 1;
5.           else
6.             return(n * factorial(n-1));
7.     }
8.
9.  public static void main(String[] args) {
10. System.out.println("Factorial of 5 is: "+factorial(5));
11. }
12. }
```

Output:

```
Factorial of 5 is: 120
```

Java Recursion Example 4: Fibonacci Series

```
1.  public class RecursionExample4 {
2.     static int n1=0,n2=1,n3=0;
3.      static void printFibo(int count){
4.         if(count>0){
5.              n3 = n1 + n2;
6.              n1 = n2;
7.              n2 = n3;
8.              System.out.print(" "+n3);
9.              printFibo(count-1);
10.         }
11.     }
```

```
12.
13. public static void main(String[] args) {
14.    int count=15;
15.      System.out.print(n1+" "+n2);//printing 0 and 1
16.      printFibo(count-2);//n-2 because 2 numbers are already printed
17. }
18. }
```

Output:

```
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

Wrapper class in Java

Wrapper class in java provides the mechanism to convert primitive into object and object into primitive.

Since J2SE 5.0, **autoboxing** and **unboxing** feature converts primitive into object and object into primitive automatically. The automatic conversion of primitive into object is known as autoboxing and vice-versa unboxing.

The eight classes of **java.lang** package are known as wrapper classes in java. The list of eight wrapper classes are given below:

Primitive Type	Wrapper class
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long

float	Float
double	Double

Wrapper class Example: Primitive to Wrapper

```
1. public class WrapperExample1{
2. public static void main(String args[]){
3. //Converting int into Integer
4. int a=20;
5. Integer i=Integer.valueOf(a);//converting int into Integer
6. Integer j=a;//autoboxing, now compiler will write Integer.valueOf(a) internally
7.
8. System.out.println(a+" "+i+" "+j);
9. }}
```

Output:

```
20 20 20
```

Wrapper class Example: Wrapper to Primitive

```
1. public class WrapperExample2{
2. public static void main(String args[]){
3. //Converting Integer to int
4. Integer a=new Integer(3);
5. int i=a.intValue();//converting Integer to int
6. int j=a;//unboxing, now compiler will write a.intValue() internally
7.
8. System.out.println(a+" "+i+" "+j);
9. }}
```

Output:

```
3 3 3
```