The switch statement

Java has a built-in multiway decision statement known as a switch. The switch statement tests the value of a given variable(or expression) against a list of case values and when a match is found, a block of statements associated with the case is executed.

Syntax :-

```
switch (expression )
   {
 case value-1:
       block-1;
        break;
  case value-2:
       block-2;
       break;
…….
……..
default :
     default block;
     break;
……
}
Statement-x;
```

WAP to enter number and print corresponding day of week using switch….case

```
import java.io.*;
class day
  {
  public static void main(String args[]) throws IOException
   {
    int num;
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  System.out.println("enter number :");
  num=Integer.parseInt(br.readLine());
switch(num)
{
case 1:
   System.out.println(" Monday");
   break;
case 2:
   System.out.println(" Tuesday");
   break;
case 3:
   System.out.println(" Wednesday");
   break;
case 4:
   System.out.println(" Thrusday");
   break;
```

```
case 5:
  System.out.println("Friday");
  break;
case 6:
  System.out.println(" Saturday");
  break;
case 7:
  System.out.println("Sunday");
  break;
default:
  System.out.println(" Please enter number between 1 to 7");
  break;
  }
 }
}
```

The ?: operator

The Java language has an unusual operator, useful for making two-way decisions. This operator is a combination of ? and : and takes three operands. This operator is popularly known as the conditional operator.

Syntax :

Conditional expression ? exp1:exp2

The conditional expression is evaluated first. If the result is nonzero, exp1 is evaluated. Otherwise exp2 is evaluated.

ex : if(x<0) flag=0;
flag=(x<0)? 0:1;
else
flag=1;

Switch case and nested ifs	**Nested if**
a) The switch() can only test for equality relational i.e only constant values are applicable.	a) The if can evaluate or logical expressions.
b) No two case statements have identical constants in the same switch.	b) Same conditions may be repeated for number of times.
c) Character constants are automatically converted to integers.	c) Character constants are automatically converted to integers.
d) If switch() case statement nested if can be used.	d) In nested if statement switch() case can be used.

Loop Control Statements

Loop :- A loop is defined as a block of statements which are repeatedly executed for certain number of times.

Java language provides for three constructs for performing loop operations :-

1) The while statement
2) The do statement
3) The for statement

The while statement

The simplest of all the looping structures in c is the while statement.

Syntax :-

```
initialization;
while (test condition)
  {
  body of the loop;
  increment/decrement;
  }
```

WAP to print 1…10 using while loop

```
class print
 {
 public static void main(String args[])
  {
   int i=1;
   while(i<=10)
     {
     System.out.println(i);
     i++;
     }
  }
 }
```

The for statement

The for loop is an entry-controlled loop that provides a more concise loop control structure.

Syntax :-

```
for (initialization;test-condition;incr/decr)
  {
  body of the loop;
  }
```

```
class loop1
{
public static void main(String args[])
 {
 int i;
```

```
 for(i=1;i<=10;i++)
   {
    System.out.println(i);
   }
  }
  }
```

The do statement

```
initialization;
do
  {
 body of the loop;
 incr/decr;
}while(condition);

class loop2
{
public static void main(String args[])
 {
 int i=1;
do
 {
 System.out.println(i);
 i ++;
}while(i<=10);
```

The for loop can be specified by different ways

1) for(;;) -> infinite loop -> No arguments
2) for(a=0;a<=20;) -> infinite loop -> 'a' is neither increased nor decreased
3) for(a=0;a<=10;a++) ->Display 1 to 10 -> 'a' is increased from 0 to 10.
   System.out.println(a);
4) for(a=10;a>=0;a--) -> Displays value -> 'a' is decreased from 10 to 0
   System.out.println(a); from 10 to 0

Note : The initialization section has two parts p=1 and n=1 separated by a comma.

Ex-
```
for(n=1,m=50;n<=m;n=n+1,m=m-1)
        {
        p=m/n;
        System.out.println(n,m,p);
        }
```
The multiple arguments in the increment section are separated by commas.

Note : It is also the test-condition may have any compound relation and the testing need not be limited only to the loop control variables.

e.g,
```
sum=0;
for(i=1;i<20  && sum<100; ++i)
```