Class- X , Computer Applications

Lesson - 8

Notes and Exercise programs of L-8

Control Flow and Decision Making in Java

Java is a structured programming language and Java program statements can be executed sequentially, conditionally (with the help of if-else, and switch), iteratively (with the help of loops) or by following a combination of all depending upon the program logic. Conditional and iterative execution of program statements is done by control flow statements. Conditional statements in Java include if, else, switchand case statements, while iterative statements include for(), while(), and do...while() statements.

Decision Making in Java

To facilitate conditional control flow in Java there are relational and logical operators to form a conditional expression that returns either true or false. Words true and falselook like keywords but they are boolean literals actually and cannot be used as identifiers in Java programs. Java program decides the execution path on basis of the truth or falsehood of conditional expression.

Java Statements and Blocks

In a Java program zero or more statements enclosed in a pair of curly braces({ }) make a **block** of statements. A block is treated as a single unit and can be used where a single statement is allowed. In fact, **a block** is a statement but this **is a compound statement**. We will soon come to know the worth of a block when we will have to control more than one statement by decision making constructs of Java.

Java If Statement

An if statement is the most basic Java control flow statement you will see in Java programs along with an optional else part. Following is the general **syntax** of ifstatement:

```
if (booleanExpression)
statement-1;

 OR

if (booleanExpression)
{
      statement-1;
      statement-2;
       . . . statement-n;
}
```

The booleanExpression in parentheses must return a

boolean true or false or Boolean (boolean wrapper). The expression can be a relational or logical expression or a function call that returns a boolean literal. Above syntax will execute statement-1 to statement-n if the booleanExpression in parentheses returns true, nothing otherwise.

By default if controls only one statement, therefore if you wish to control only one statement by if, you need not to enclose the only statement within curly braces. But, **it is good to use braces every time with ifbecause it increases readability** of the program. If there are two or more statements to be controlled by ifconditional, they all must be enclosed in curly braces. A set of statements enclosed in braces is called a **block or a compound statement**. For demonstration, consider the following example program:

```
//Demonstrates if-else statement
 public class ControlFlowDemo
{
public static void main(String[] args)
{
int x = 10, y = 20;
boolean decision = false;   //if controls one statement by default, so no braces required
if(x < y)
System.out.println(x + " is less than " + y + "\n");
 if (decision==true)
System.out.println("always false\n");
```

```
//will never be printed
if (isPositive(x)==true)
System.out.println(x + " is positive: " + isPositive(x) +
"\n");
 if (x > y)
{
System.out.println("Within from if block");
System.out.println(x + " greater than " + y + "\n");
}
 else
{
System.out.println("Within from else block");
System.out.println("Optional else used when there are
two branches"); System.out.println(x + " less than " + y
+ "\n");
}
 //will print "yes, variable decision is false" if (decision
== false) System.out.println("Yes, variable decision is
false\n");
 //what do you think it will print?
if (decision = true) System.out.println("Variable
decision is assigned to true");
}   // number is positive if it is greater than zero
 public static boolean isPositive (int n1)
 {
return (n1 > -1);
 }
}
```

OUTPUT ====== 10 is less than 20 10 is positive: true Within from else block Optional else used when

there are two branches 10 less than 20 Yes, variable decision is false Variable decision is assigned to true

In above example code, look at the second if statement if (decision), where variable decision returns false, therefore the statement System.out.println("always false\n"); will not be executed and nothing will be printed.

Next, in if (decision == false) statement, variable decision is being checked for equality against boolean literal false. As variable decision contains false so it will pass the equality test and will return true.

Finally, in the last most if statement decision = true is not a valid relational or logical expression, while it is a mere assignment operation and boolean literal true is being assigned to variable decision. But, still there is no error and code is executed successfully because this is eventually evaluated to true.

Java Nested If Statement

When an if statement appears inside the other it is called nesting of if statements. Nesting of if statements is very helpful when you have something to do by following more than one decision. For example, if you meet your daughter's school teacher every second Saturday of the month to get to

know the performance of your doll then your meeting is followed by two decisions. First, it should be a Saturday then it should be the second of the month. And more practically, the third one is that it should not be a holiday. Let's program your meeting schedule as follows:

```
boolean isSat = true; int whichSat = 2; boolean
isHoliday = false;
  if (isSat==true)
{
      if (whichSat == 2)
      {
          if (isHoliday == false)
          {
              System.out.println("It is meeting today.");
           }
       }
}
else
{
 System.out.println("No meeting today.");
 }
```

While nesting if statements we must know that an else statement is always bound to its closest if. Following piece of code demonstrates that:

```
int i = 10, j = 15, k = 50, a = 5, b = 7, c = 9, d = 11;
 if(i == 10)
```

```
{
    if(j < 20)
       a = b;
    if(k > 100)
       c = d;
    else
        a = c; // associated with if(k > 100)
}
 else
    a = d; // associated with if(i == 10)
System.out.println(a);
```

OUTPUT ====== 9

Java If-else if Ladder

Java control flow statements are executed from top to down, therefore, a ladder of if-else conditions will be evaluated from top to down. As soon as an if statement from the ladder evaluates to true, the statements associated with that if are executed, and the remaining part of the ladder is bypassed. The last most else is executed only when no condition in the whole ladder returns true.

Here is a program which demonstrates if-else ladder. It determines if a given alphabet is vowel or consonant.

```
//Demonstrates if-else ladder
public class ControlFlowDemo
```

```
{
public static void main( )
{
 char ch = 'o';
 if (ch == 'a' || ch == 'A') System.out.println(ch + " is
vowel.");
 else if (ch == 'e' || ch == 'E') System.out.println(ch + " is
vowel.");
else if (ch == 'i' || ch == 'I') System.out.println(ch + " is
vowel.");
 else if (ch == 'o' || ch == 'O') System.out.println(ch + " is
vowel.");
else if (ch == 'u' || ch == 'U') System.out.println(ch + " is
vowel.");
else
 System.out.println(ch + " is a consonant.");
}
 }
```

OUTPUT ====== o is vowel.

In above program, one and only one printInstatement will be executed, no matter what is the value of ch from a-z or A-Z.

switch....case

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked

for each case.
switch case statement is generally used for writing menu driven programs or for users choice programs.
It's execution is faster than if...else if...else ladder.
Write the differences between if... else and switch statements.

Syntax

The syntax of enhanced for loop is −

```
switch(expression)
 {
case value :
    // Statements
     break; // optional
case value :
    // Statements
    break; // optional

// You can have any number of case statements.
   default : // Optional
  // Statements
 }
```

The following **rules** apply to a **switch** statement −

- Duplicate case values are not allowed.

- The variable used in a switch statement can only be integers, convertable integers (byte, short, char),

strings and enums.

Beginning with JDK7, it also works with enumerated types (Enums in java), the String class and Wrapperclasses.

- [ ] You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.

- [ ] The value for a case must be the same data type as the variable in the switch and it must be a constant or a literal.

- [ ] When the variable being switched on is equal to a case, the statements following that case will execute until a break statement is reached.

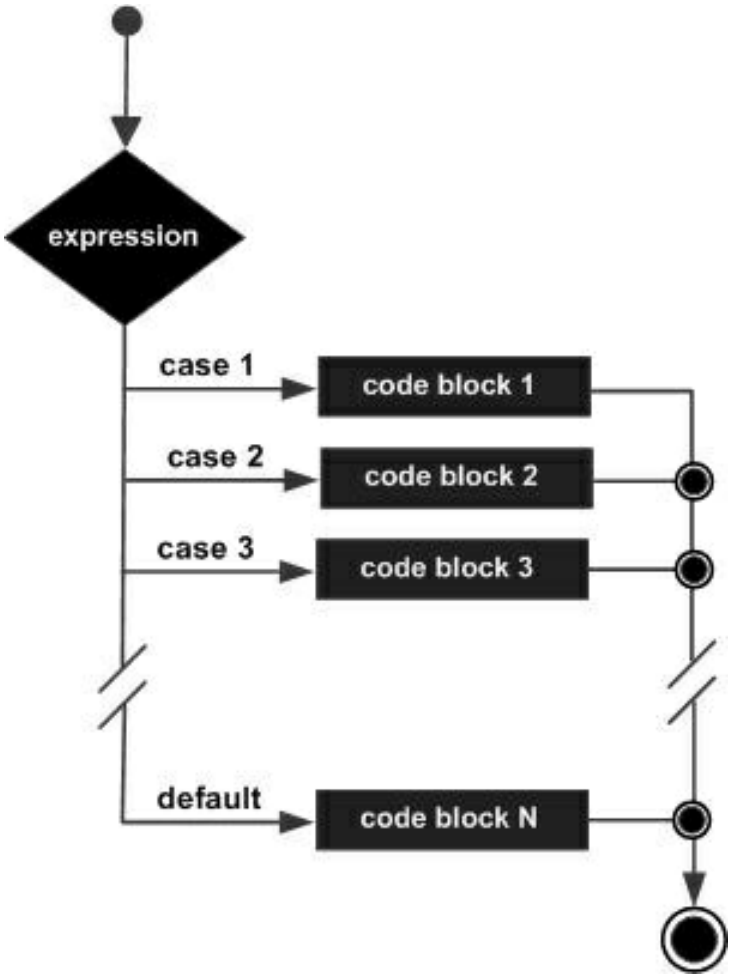
- [ ] When a break statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement.The break statement is used inside the switch to terminate a statement sequence. That's why it is also called case terminator.

- [ ] Not every case needs to contain a break. If no break appears, the flow of control will fall through to subsequent cases until a break is

reached.

- A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. No break is needed in the default case.

Flow Diagram



Example 1:

```
public class Test
 {
      public static void main(String args[])
      {
       // char grade =args[0].charAt(0);         char grade =
'C';
switch(grade)
{
    case 'A' : System.out.println("Excellent!");
break;
case 'B' :
case 'C' :
  System.out.println("Well done");
   break;
case 'D' :
  System.out.println("You passed");
 case 'F' :
  System.out.println("Better try again");
  break;
default :
  System.out.println("Invalid grade");
 }
 System.out.println("Your grade is " + grade);
}
}
```

Compile and run the above program using various command line arguments. This will produce the

following result –

Output

Well done Your grade is C

Example 2:

```
// Java program to demonstrate switch
// case with multiple cases and
// break statements

public class Test
{
public static void main(String[] args)
  {
   int day = 2;
   String dayType;
   String dayString;
   switch (day)
    {
         case 1:

           dayString = "Monday";

           break;

       case 2:

           dayString = "Tuesday";
```

```
    break;
case 3:
    dayString = "Wednesday";
    break;
case 4:
    dayString = "Thursday";
    break;
case 5:
    dayString = "Friday";
    break;
case 6:
    dayString = "Saturday";
    break;
case 7:
    dayString = "Sunday";
```

```
            break;

        default:

            dayString = "Invalid day";

        }
System.out.println(dayString);
}
```

Output :
Tuesday

Example 3:

```
// Java program to demonstrate switch
// case with multiple cases without
// break statements
public class ExampleOfFallthrough
{

    public static void main( )
    {

        int day = 2;

        String dayType;

        switch (day)
        {
```

```
        // multiple cases without break
        //statement
        case 1:

        case 2:

        case 3:

        case 4:

        case 5:

            dayType = "Weekday";

            break;

        case 6:

        case 7:

            dayType = "Weekend";

            break;
      }
System.out.println(day + "day is a " + dayString + ".");
}
```

Output:

2 day is a Weekday

Nested Switch Case statements

We can use a switch as part of the statement sequence of an outer switch. This is called a nested switch. Since a switch statement defines its own block, no conflicts arise between the case constants in the inner switch and those in the outer switch. For example:

```
// Java program to demonstrate

// nested switch case statement

public class Test
 {

   public static void main(  )

   {

      String Branch = "CSE";

      int year = 2;
      switch (year)
      {
      case 1:

         System.out.println("elective courses : Advance
```

```
english, Algebra");

            break;

        case 2:

            switch (Branch) // nested switch
           {

            case "CSE":

            case "CCE":

                System.out.println("elective courses : Machine
Learning, Big Data");

                break;
             case "ECE":

                System.out.println("elective courses : Antenna
Engineering");

                break;
            default:
                System.out.println("Elective courses :
Optimization");

            } //end of nested switch
default :
   System.out.println("Year must be 1 or 2 only.....");
```

```
        } // end of outer switch
    }
}
```

Output:

elective courses : Machine Learning, Big Data

Exercise programs
With each program write comments and Variable Description

Q17

```
class Numbers
{
public static void main(  )
 {
 int sum=0;
System.out.println(" Numbers divisible by 17 are:");
for(int i=1; i<=1000;i++)
{
if( i%17==0)
{
System.out.println(i);
sum=sum+i;
}
}
}
System.out.println("Sum of the numbers which are divisible by 17 is :" + sum);
}
}
```

Variable Description

Name	Data Type	Description
sum	int	to store the sum of the numbers divisible by17
i	int	control variable of for() loop

Q(18)

```
import java.util.Scanner;
class Digits
{
  public static void main( )
  {
    Scanner sc= new Scanner( System.in);
     int n=0;
    System.out.println( " Enter a number ");
     n=sc nextInt( );
     if(n>=10 && n<=99)
     System.out.println( n+" is a two digits number. ");
      else if(n>=100 && n<=999)
       System.out.println( n+" is a three
 digits number. ");
      else if(n>=1000 && n<=9999)
       System.out.println( n+" is a four
 digits number. ");
}
}
```

Q20

```
import java.util.Scanner;
class DescendingNumbers
{
   public static void main( )
   {
      Scanner sc= new Scanner( System.in);
       int a=0,b=0,c=0;
      System.out.println( " Enter  three number ");
       a=sc nextInt( );
        b=sc nextInt( );
       c=sc nextInt( );
System.out.println( " The numbers in descending order
are:");
if(a>b)
{
   if(a>c)
  {
      if(b>c)
          System.out.println(a+" ,"+b+" ,"+c);
      else
          System.out.println(a+" ,"+c+" ,"+b);
   }
    else
   System.out.println(c+" ,"+a+" ,"+b);
}
else
{
       if(b>c)
```

```
      {
          if(a>c)
            System.out.println(b+" ,"+a+" ,"+c);
     else
        System.out.println(b+" ,"+c+" ,"+a);
      }
      else
           System.out.println(c+" ,"+b+" ,"+a);
   }
}
}
```

Q (22)

```
public class MonthName
{
public static void main(int month)
 {
       String mString;
      switch (month)
       {
          case 1:
             mString = "January";
             break;
          case 2:
             mString = "February";
             break;
           case 3:
              mString = "March";
              break;
```

```
            case 4:
                mString = "April";
                break;
            case 5:
                mString = "May";
                break;
             case 6:
                mString = "June";
                break;
              case  7 :
                 mString = " July";
                 break;
               case   8:
                  mString = " August";
                  break;
               case   9 :
                    mString = " September";
                    break;
              case   10 :
                    mString = " October";
                    break;
               case   11 :
                     mString ="November";
break;
                case   12 :
                     mString = "December ";
                    break;
                default:
                     mString = "Invalid month.....";
    }
```

```
System.out.println(mString);
}
```

Q24

```
import java.util.Scanner;
class CityName
{
  public static void main( )
  {
    Scanner sc= new Scanner( System.in);
    char ch=0;
System.out.println(" Enter city character as : \n Delhi - D\n Mumbai - M \n Kolkata -  K \n Chennai - C");
System.out.println(" Enter your choice D/M / K / C :");
ch= sc.next( ).charAt(0);
switch(ch)
{
case 'D' :
System.out.println(" Delhi");
break;
case 'M' :
System.out.println(" Mumbai");
break;
case 'K' :
System.out.println(" Kolkata");
break;
case 'C' :
System.out.println(" Chennai");
break;
```

```
default :
System.out.println(" Wrong choice...");
}
}
}
```

Q30

```
import java.util.Scanner;
class ElectricityBill
{
   public static void main( )
   {
      Scanner sc= new Scanner( System.in);
       int units=0;
       float bill=0.0f;
      System.out.println( " Enter  the total number of
units ");
       units=sc nextInt( );
       if(units>=1 && units<=100)
          bill=units×.80f;
        else if(units>=101 && units<=300)
          bill=(100×.80f) +(( units-100)×1);
        else if(units>300)
          bill=(100×.80f) + (200×1)+(( units-300)×2.50f);
          bill=bill+500;
 System.out.println("Units are : "+units);
System.out.println("Bill is : "+ bill);
}
}
```

Q31

```
import java.util.Scanner;
class CheckNumber
{
  public static void main( )
  {
    Scanner sc= new Scanner( System.in);
     int ch=0,num=0;
System.out.println(" (1) Automorphic number \(2)
Buzz number ");
System.out.println(" Enter your choice 1/2 ");
ch=sc.nextInt( );
System.out.println(" Enter a number" );
num= sc.nextInt( );

switch(ch)
{
   case 1:
     int c=0,rem=0, s=0;
    s=num × num;
    int t=num;
     while(t>0)
     {
        c++;   // counting the digits in no.
         t=t/10;
      }
       rem=s%(int)Math.pow(10,c);
       if(rem==num)
          System.out.println(num + " is an Automorphic
```

```
no.");
        else
             System.out.println(num + " is NOT an
Automorphic no.");
        break;
     case 2:
         if(num%10==7 || num%7==0)
          System.out.println(num + " is a BUZZ no.");
         else
           System.out.println(num + " is NOT a BUZZ no.");
break;
default:
System.out.println(" Wrong choice.... ");
}
}
}
```

Q35

```
import java.util.Scanner;
class TotalAmount
{
   public static void main( )
   {
      Scanner sc= new Scanner( System.in);
       int dis=0;
       float tamt=0.0f, cost=0f;
       char ch=0;
System.out.println(" Enter the item code as : \n Laptop-
L\n LCD - D \n XBox - X \n Printer - P");
ch= sc.next( ).charAt(0);
```

```
      System.out.println( " Enter  the cost  of the item");
       cost=sc.nextFloat( );
       if  (ch=='L')
          dis=5;
       else    if(ch=='D')
          dis=7;
       else    if(ch=='X')
          dis=10;
       else    if(ch=='P')
          dis=11;
        else
            System.out.println(" Wrong item code");
   dis=cost×dis/100f;
    tamt=cost-dis;
System.out.println("Cost of the item is :"+ cost);
System.out.println("Item code is :"+ ch);
System.out.println("Discount amount is :"+ dis);
System.out.println("Payable amount after getting
discount  is :"+ tamt);
}
}
```

Q37

```
import java.util.Scanner;
class ClothDiscount
{
   public static void main( )
   {
      Scanner sc= new Scanner( System.in);
```

```
      float len=0.0f, dis=0f,tamt=0f,    paidAmt=0f;;
      char ch=0;
System.out.println(" Enter   D - Dealer\n R -  Retailer");
System.out.println("Enter your choice ( D/R)  :");
ch= sc.next( ).charAt(0);
System.out.println("Enter the length of the cloth :");
len=sc.nextFloat( );
System.out.println("Enter the total amount of purchase
");
tamt=sc.nextFloat( );

switch(ch)
{
case 'D' :
if(len>=1 && len<=1000)
    dis=20;
else if(len>1000 && len<=2000)
    dis=25;
else if(len>2000)
    dis=35;
dis=tamt×dis/100;
paidAmt=tamt-dis;
break;

case 'R':
if(len>=1 && len<=1000)
    dis=15;
else if(len>1000 && len<=2000)
    dis=20;
```

```
else if(len>2000)
    dis=25;
dis=tamt×dis/100;
paidAmt=tamt-dis;
break;
default:
System.out.println(" Wrong customer type....");
}
// output
System.out.println("Type is customer is :"+ch);
System.out.println("Length of the cloth is :"+len);
System.out.println("Discount is :"+dis);
System.out.println("Amount to be paid after
calculating the discount : "+paidAmt):
}
}
```

Q(38)

```
import java.util.Scanner;
class MoneyConversion
{
  public static void main( )
  {
    Scanner sc= new Scanner( System.in);
     int ch=0,rs=0, dollar=0;

System.out.println("****** MENU ******");
System.out.println(" (1) Rupees into Dollar number \(2)
Dollar into Ruprrs");
```

```
System.out.println(" Enter your choice 1/2 ");
ch=sc.nextInt( );

switch(ch)
{
   case 1:
System.out.println(" Enter rupees :");
rs= sc.nextInt( );
dollar=rs/77;
System.out.println(" Rupees are : "+rs);
System.out.println(" After conversion
 Dollars are :"+dollar);
break;

case 2:
System.out.println(" Enter dollars :");
dollar= sc.nextInt( );
rs=dollar×77;
System.out.println(" Dollars are : "+dollar);
System.out.println(" After conversion
 Rupees are :"+rs);
break;

default:
System.out.println(" Wrong  choice....");
}
}
}
```

Last modified: 9:23 am