Darshan
UNIVERSITY
योग: कर्मसु कौशलम्

Unit 2: Selections, Mathematical functions and loops

Prof. Swati R Sharma

Computer Engineering Department

Darshan Institute of Engineering & Technology, Rajkot
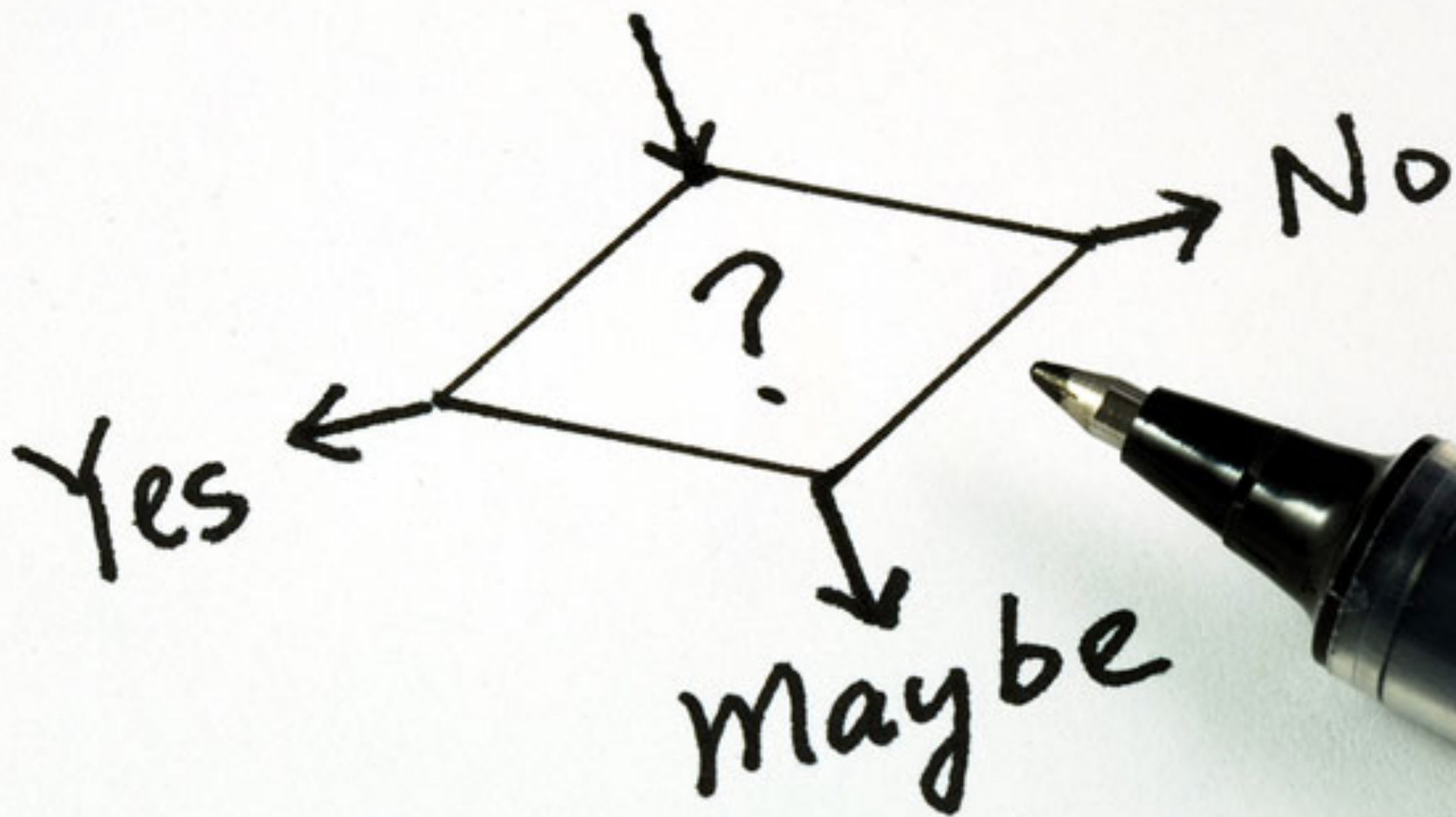
✉ swati.sharma@darshan.ac.in

☎

✅ What we will learn

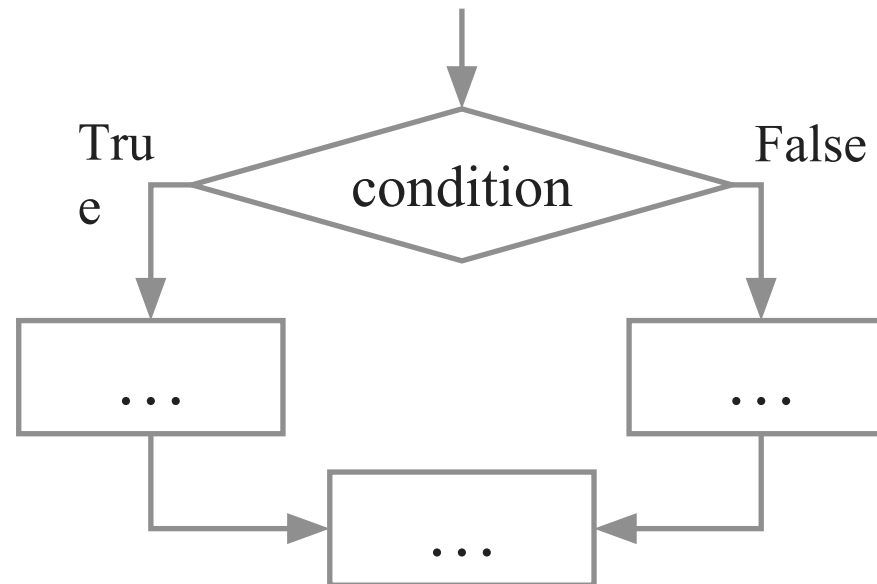- ✔ If statement
- ✔ Two way if statement
- ✔ Nested if statement
- ✔ Switch statement
- ✔ Conditional Expression
- ✔ While loop
- ✔ Do-while loop
- ✔ For loop
- ✔ Nested loop
- ✔ Break and continue statement
- ✔ Common mathematical expression

Unit-2

Darshan UNIVERSITY

Decision Making

 Compiler executes program statements sequentially.

 Decision making statements are used to control the flow of program execution.

 It allows us to control whether a set of program statement should be executed or not.

 It evaluates condition or logical expression first and based on its result (true or false), the control is transferred to the particular statement.

 If result is true then it takes one path else it takes another path.

Decision Making Statements

 Commonly used decision making statements are

  One way Decision:	**if**	(Also known as simple if)

  Two way Decision:**if**…**else**

  Multi way Decision:	**if**…**else if**…**else if**…**else**

  Decision within Decision: **nested if**

  Two way Decision:**?: (Conditional Operator)**

  n-way Decision:	**switch**…**case**

if

One way Decision

if

- **if** statement is the most simple decision-making statement also known as **simple if**.

- An **if** statement consists of a boolean expression followed by one or more statements.

- **If** the expression is true, then 'statement-inside' will be executed, otherwise 'statement-inside' is skipped and only 'statement-outside' will be executed.

- It is used to decide whether a block of statements will be executed or not i.e if a certain condition is true then a block of statement is executed otherwise not.

condition — False

Tru
e
statement-inside
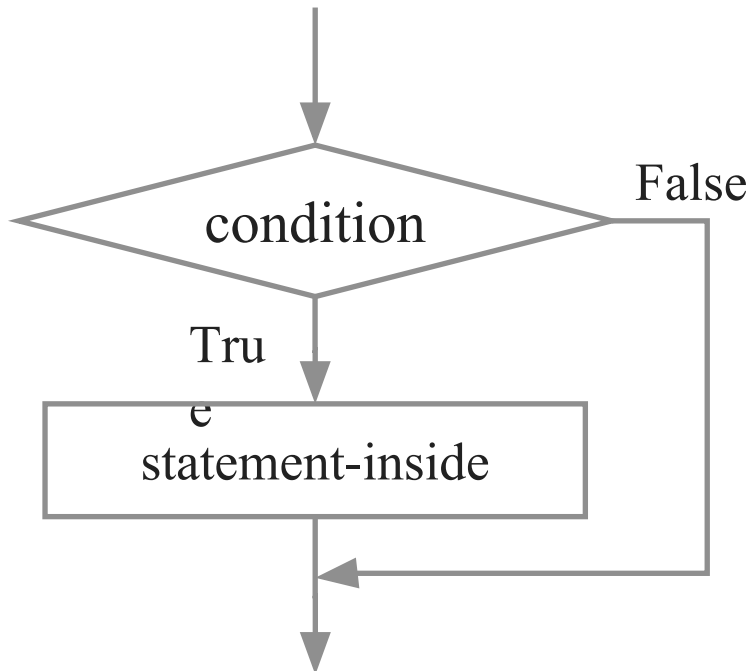
```
if(condition)
{
    // Statements to execute if condition is true
}
```

Darshan
UNIVERSITY

WAP to print if a number is positive

```java
1. import java.util.*;
2. class MyProgram{
3. public static void main (String[] args){
4. int x;
5. Scanner sc = new Scanner(System.in);
6.     x = sc.nextInt();
7.     if(x > 0){
8.        System.out.println("number is a positive");
9.     }
10. }
```

Exercise

☐ Write a program which reads two numbers and based on different between it prints either of following message DIFFERENCE IS POSITIVE or DIFFERENCE IS NAGATIVE.

WAP to print if a number is odd or even

```java
1. import java.util.*;
2. class MyProgram{
3. public static void main (String[] args){
4. int x;
5. Scanner sc = new Scanner(System.in);
6.     x = sc.nextInt();
7.     if( x % 2 == 1 ){
8.         System.out.println("number is a odd");
9.     }
10.     if( x % 2 == 0 ){
11.         System.out.println("number is a even");
12.     }
13. }
```
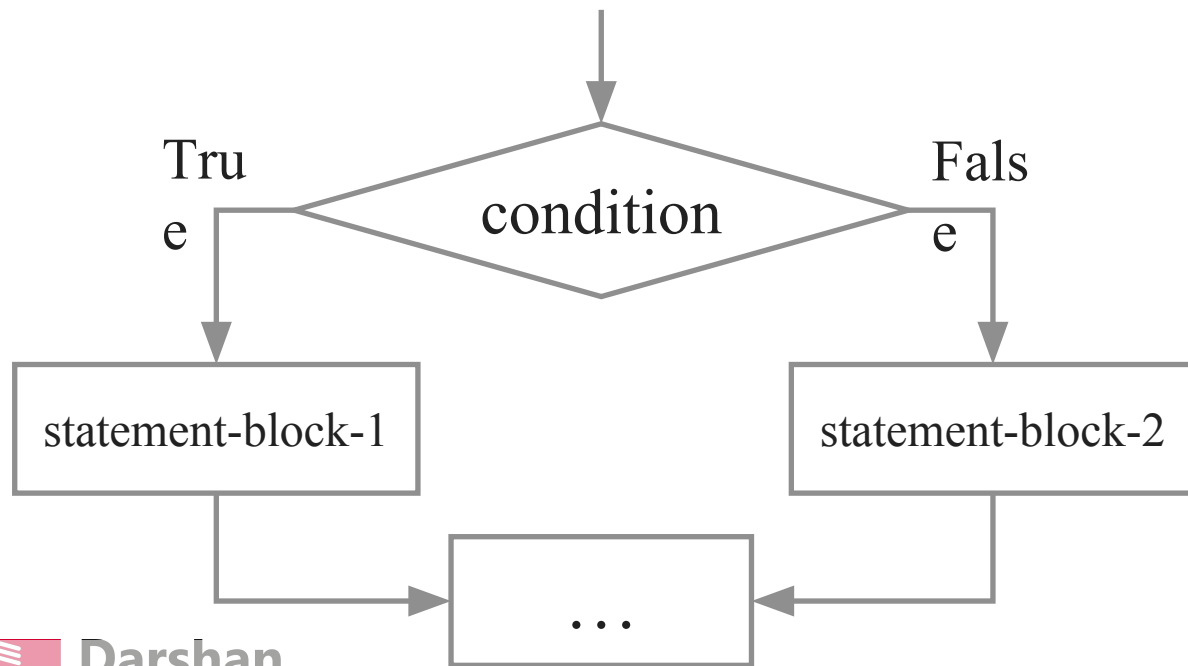
if…else

Two way Decision

If…else

- For a **simple if**, if a condition is true, the compiler executes a block of statements, if condition is false then it doesn't do anything.

- What if we want to do something when the condition is false? if…else is used for the same.

- If the 'expression' is true then the 'statement-block-1' will get executed else 'statement-block-2' will be executed.


```
if(condition)
{
    // statement-block-1
    // to execute if condition is
    true
}
else
{
    // statement-block-2
    // to execute if condition is
    false
}
```

WAP to print if a number is positive or negative

```java
1.  import java.util.*;
2.  class MyProgram{
3.  public static void main (String[] args){
4.  int x;
5.      Scanner sc = new Scanner(System.in);
6.      x = sc.nextInt();
7.      if (x > 0){
8.          System.out.println("Number is positive");
9.      }//if
10.     else{
11.         System.out.println("Number is negative");
12.      }//else
13.     }//main
14. }//class
```

WAP to print if a number is odd or even

```java
1. import java.util.*;
2. class MyProgram{
3. public static void main (String[] args){
4.  int x;
5.  Scanner sc = new Scanner(System.in);
6.     x = sc.nextInt();
7.     if( x % 2 == 1 ){
8.        System.out.println("number is a odd");
9.     }
10.    else{
11.       System.out.println("number is a even");
12.    }
13. }
```

Exercise

1. Any year is entered through the keyboard, write a program to determine whether the year is leap or not.

2. Write a program to check whether a triangle is valid or not, when the three angles of the triangle are entered through the keyboard. A triangle is valid if the sum of all the three angles is equal to 180 degrees.

if…else if…else

Multi way Decision

if…else if…else

- **if…else if…else** statement is also known as **if-else-if ladder** which is used for multi way decision making.

- It is used when there are more than two different conditions.

- It tests conditions in a **sequence**, from top to bottom.

- If first condition is true then the associated block with if statement is executed and rest of the conditions are skipped.

- If condition is false then then the next if condition will be tested, if it is true then the associated block is executed and rest of the conditions are skipped. Thus it checks till last condition.

- Condition is tested only and only when all previous conditions are false.

- The last else is the default block which will be executed if none of the conditions are true.

- The last else is not mandatory. If there are no default statements then it can be skipped.

```
if(condition 1)
{
        statement-block1;
}
else if(condition 2)
{
    statement-block2;
}
else if(condition 3)
{
    statement-block3;
}
else if(condition 4)
{
        statement-block4;
}
else

    default-statement;
```

if-else-if ladder

```
if(condition 1)
{
    statement-block1;
}
else if(condition 2)
{
    statement-block2;
}
else if(condition 3)
{
    statement-block3;
}
else if(condition 4)
{
    statement-block4;
}
else
    default-statement;
```
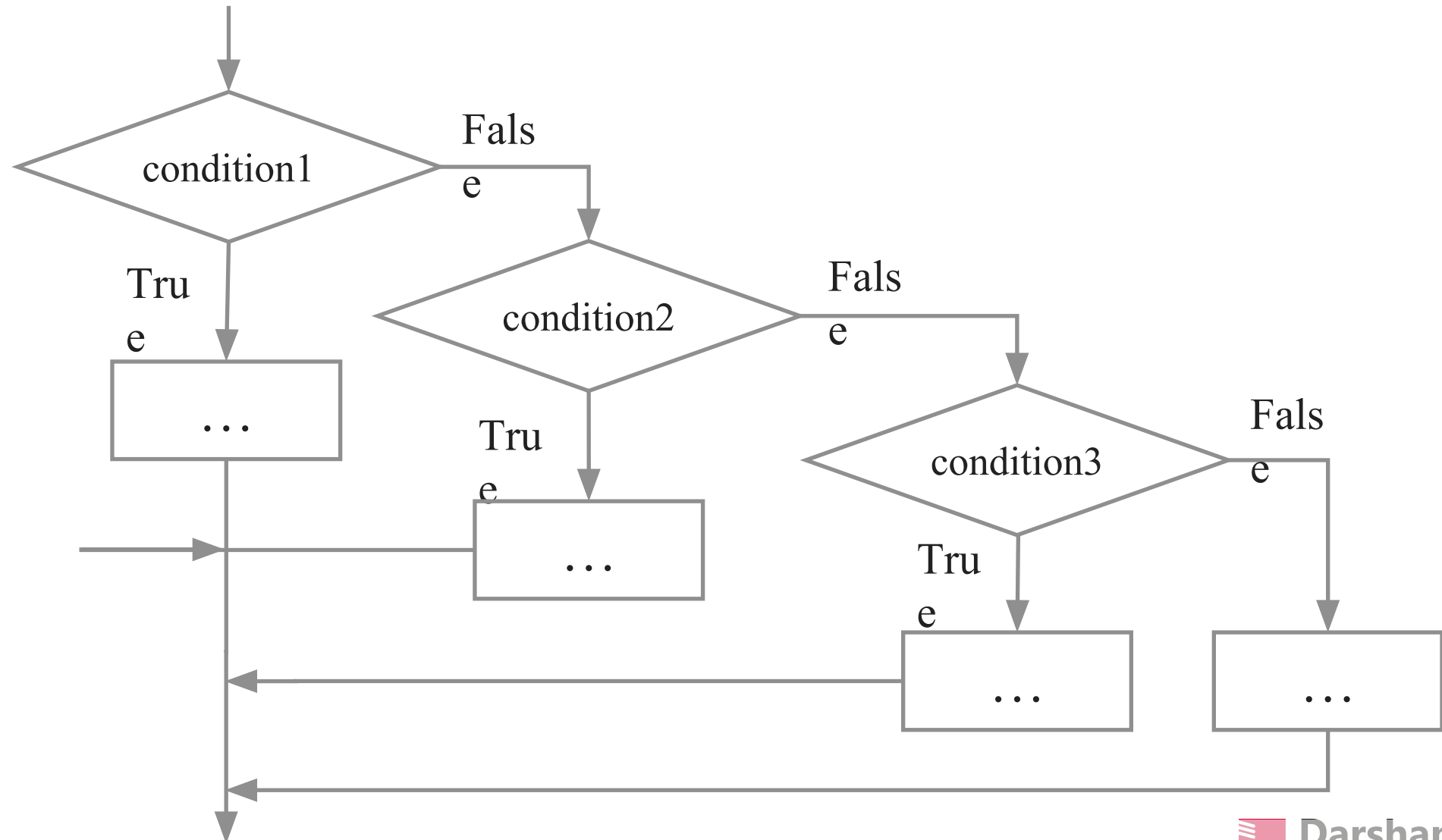
WAP to print if a number is zero or positive or negative

```java
1. import java.util.*;
2. class MyProgram{
3. public static void main (String[] args){
4.  int x;
5.  Scanner sc = new Scanner(System.in);
6.      x = sc.nextInt();
7. if(x > 0){
8.     System.out.println(" number is a positive");
9.  }
10.    else if(x < 0) {
11.    System.out.println(" number is a negative");
12. }
13.    else{
14.    System.out.println(" number is a zero");
15.  }
16. }
```

WAP to print day name from day number

```
1.  public class Demo  {
2.    public static void main(String[] args)   {
3.   int d;
4.      Scanner sc = new Scanner(System.in);
5.   d = sc.nextInt();
6.   if (d == 1 )      System.out.println("Monday");
7.   else if (d == 2)    System.out.println("Tuesday");
8.   else if (d == 3)    System.out.println("Wednesday");
9.     else if (d == 4)     System.out.println("Thursday");
10.    else if (d == 5)  System.out.println("Friday");
11.    else if (d == 6)  System.out.println("Saturday");
12.    else          System.out.println("Sunday");
13.  }
14. }
```

if-else statement

```java
1.int marks = 65;
2.if (marks < 60) {
3.  System.out.println("fail");
4.} else if (marks >= 60 && marks < 80) {
5.  System.out.println("B grade");
6.} else if (marks >= 80 && marks < 90) {
7.  System.out.println("A grade");
8.} else if (marks >= 90 && marks < 100) {
9.  System.out.println("A+ grade");
10.} else {
11.  System.out.println("Invalid!");
12.}
```

Nested If

- A nested if is an if statement that is the target of another if statement.

- Nested if statements mean an if statement inside another if statement.

- The statement connected to the nested if statement is only executed when -:
  - condition of outer if statement is true, and
  - condition of the nested if statement is also true.

- Note: There could be an optional else statement associated with the outer if statement, which is only executed when the condition of the outer if statement is evaluated to be false and in this case, the condition of nested if condition won't be checked at all.

```
if(condition 1)
{
    if(condition 2)
    {
        nested-block;
    }
    else
    {
        nested-block;
    }
}//if
else if(condition 3)
{
    statement-block3;
}
else(condition 4)
{
    statement-block4;
}
```

Nested If statement

- We can also use if/else if statement inside another if/else if statement, this is known as nested if statement.

```java
int username = Integer.parseInt(args[0]);
int password = Integer.parseInt(args[1]);
double balance = 123456.25;

if(username==1234){
    if(password==987654){
        System.out.println("Your Balance is ="+balance);
    }
    else{
        System.out.println("Password is invalid");
    }
}
else{
    System.out.println("Username is invalid");
}
```

Exercise

- In a company an employee is paid as under:
  - If his basic salary is less than Rs. 1500, then HRA = 10% of basic salary and DA = 90% of basic salary.
  - If his salary is either equal to or above Rs. 1500, then HRA = Rs. 500 and DA = 98% of basic salary.
  - Employee's salary is input through the keyboard, write a program to find his gross salary.

switch...case

n-way Decision

Switch…case

- switch…case is a multi-way decision making statement.

- It is similar to if-else-if ladder statement.

- It executes one statement from multiple conditions.

```
switch (expression)
{
    case constant 1:
        // Statement-1
        break;

    case constant 2:
        // Statement-2
        break;

    case constant 3:
        // Statement-3
        break;

default:
        // Statement-default
//  if none of the above case matches
then this block would be executed.
}
```

Darshan
UNIVERSITY

Switch…case: WAP to print day based on number entered

```java
1. public class Demo  {
2.    public static void
   main(String[] args){
3. int d;
4. Scanner sc= new
Scanner(System.in);
5. d = sc.nextInt();
6. switch (d) {
7.      case 1:
8.          System.out.println("Monday"); break;
9.      case 2:
10.         System.out.println("Tuesday"); break;
11.     case 3:
12.         System.out.println("Wednesday"); break;
13.     case 4:
14.         System.out.println("Thursday"); break;
15.     case 5:
16.         System.out.println("Friday"); break;
17.     case 6:
18.         System.out.println("Saturday"); break;
19.     case 7:
20.         System.out.println("Sunday"); break;
21.     default:
22.         System.out.println("Invalid Day");
23. } //switch
24. }
25.}
```

Darshan UNIVERSITY

Switch…case

☐ switch statement executes one statement from multiple conditions. It is like if-else-if ladder statement.

```java
public class SwitchExampleDemo {
    public static void main(String[] args)
    {
        int number = 20;
        switch (number) {
            case 10:
                System.out.println("10");
                break;
            case 20:
                System.out.println("20");
                break;
            default:
                System.out.println("Not 10 or 20");
        }
    }
}
```

Exercise

 Write a menu driven program that allows user to enters five numbers and then choose between finding the smallest, largest, sum or average. Use switch case to determine what action to take. Provide error message if an invalid choice is entered.

Points to remember for switch…case

 The condition in the switch should result in a constant value otherwise it would be invalid.

 In some languages, switch statements can be used for **integer values** only.

 Duplicate case values are not allowed.

 The value for a case must be of the same data type as the variable in the switch.

 The value for a case must be a constant.

 variables are not allowed as an argument in switch statement.

 The break statement is used inside the switch to terminate a statement sequence.

 The break statement is optional, if eliminated, execution will continue on into the next case.

 The default statement is optional and can appear anywhere inside the switch block.

Exercise

 Write a Java program to get a number from the user and print whether it is positive or negative.

 Write a program to find maximum no from given 3 no.

 The marks obtained by a student in 5 different subjects are input through the keyboard.

   The student gets a division as per the following rules:

     ▪ Percentage above or equals to 60-first division
     ▪ Percentage between 50 to 59-second division
     ▪ Percentage between 40 and 49-Third division
     ▪ Percentage less than 40-fail

   Write a program to calculate the division obtained by the student.

 Write a Java program that takes a number from the user and displays the name of the weekday accordingly (For example if user enter 1 program should return Monday) .

Introduction to loop
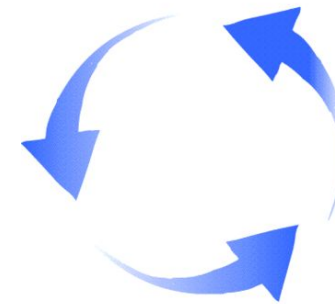
Repeatedly execute a block of statements

Loop

☐ Sometimes we need to repeat certain actions several times or till the some criteria is satisfied.

☐ Loop constructs are used to iterate a block of statements several times.

☐ Loop constructs repeatedly execute a block of statements for a fixed number of times or till some condition is satisfied

Flowchart of `if`
(`true-block` executed only once)

Flowchart of `while`
(`true-block` executed till condition is true)

condition

False

True

true-block

condition

False
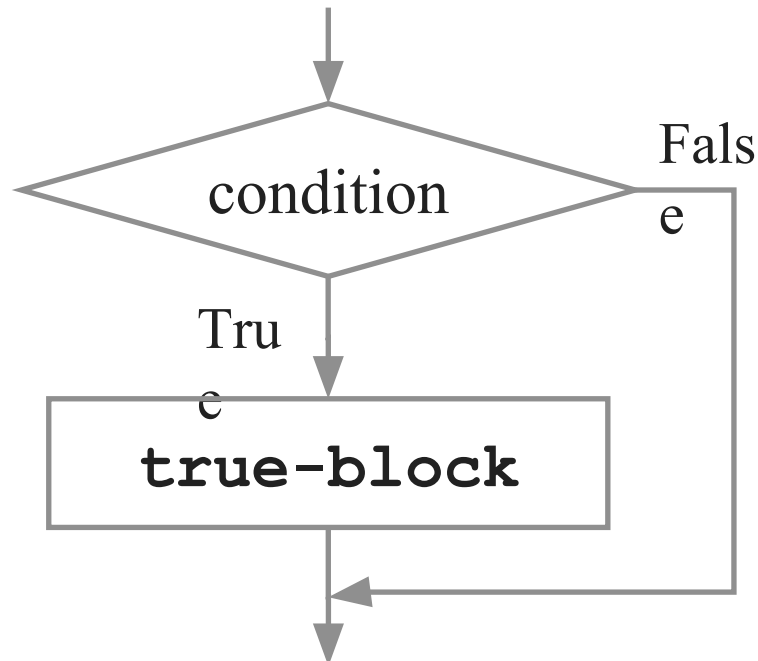
True

true-block
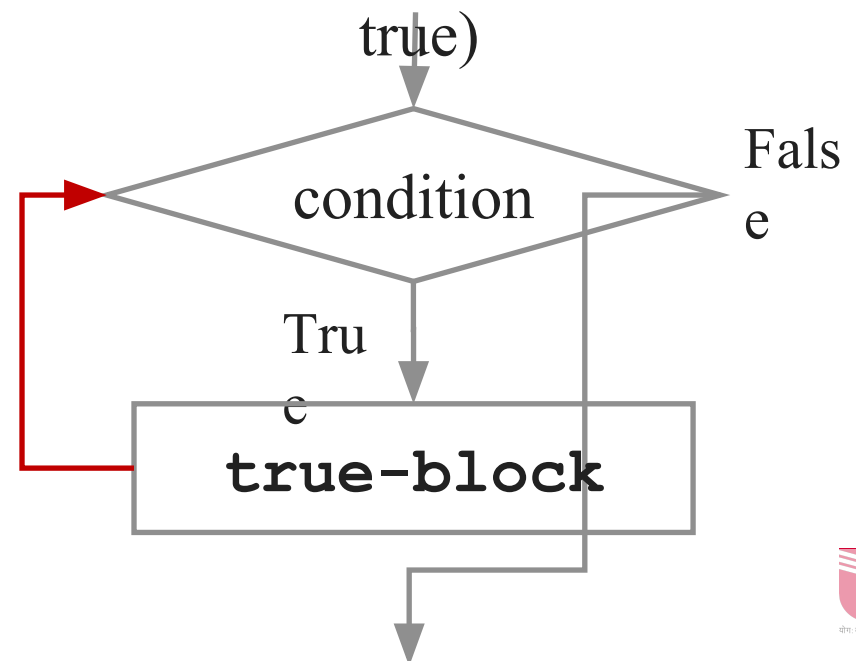
Looping Statements

☐ Following are looping statements in any programming language,

☐ Entry Controlled **`while, for`**

☐ Exit Controlled **`do`**...**`while`**

☐ Unconditional Jump **`goto`** (It is advised to never use **goto** in a program)
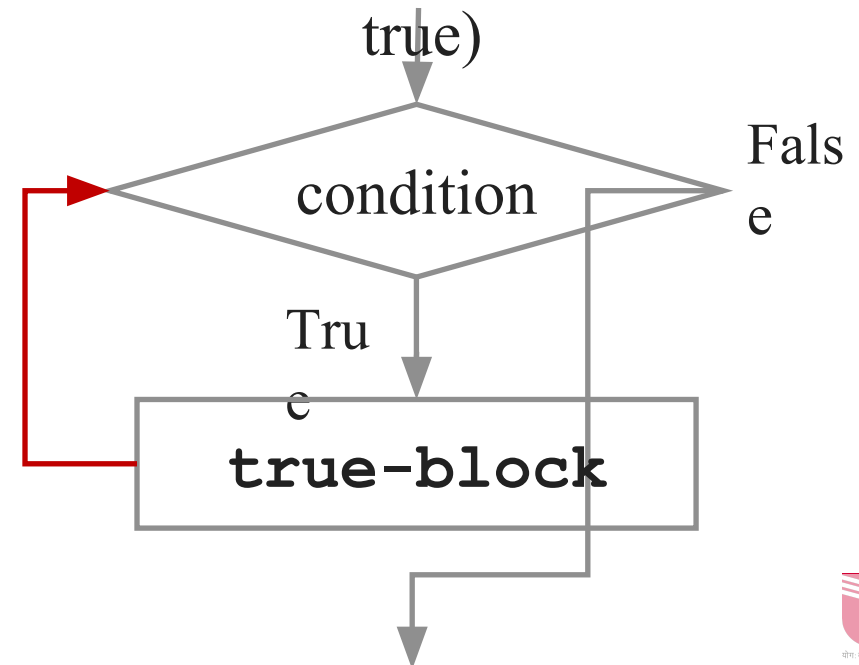
while

Entry Controlled Loop

while

- **while** is a entry controlled loop.
- It executes a block of statements till the condition is true.

```
while(condition)
{
    // true-block
}
```

```
int i = 1;
while (i <= 5)
{
    System.out.println(i);
    i++;
}
```

Flowchart of **while**
(**true-block** executed till condition is true)

While Loop

☐ while loop is used to iterate a part of the program several times. while is entry control loop.

☐ If the number of iteration is not fixed, it is recommended to use while loop.

```java
//code will print 1 to 9
public class WhileLoopDemo {
    public static void main(String[] args) {
        int number = 1;
        while(number < 10) {
            System.out.println(number);
            number++;
        }
    }
}
```

Do-while Loop

 do-while loop is executed at least once because condition is checked after loop body.

```java
//code will print 1 to 9
public class DoWhileLoopDemo {
    public static void main(String[] args) {
        int number = 1;
        do {
            System.out.println(number);
            number++;
        }while(number < 10) ;
    }
}
```

WAP to print odd numbers between 1 to n

```java
1.  import java.util.*;
2.  class WhileDemo{
3.  public static void main (String[] args){
4.   int n,i=1;
5.   Scanner sc = new Scanner(System.in);
6.   System.out.print("Enter a number:");
7.   n = sc.nextInt();
8.   while(i <= n){
9.     if(i%2==1)
10.      System.out.println(i);
11.    i++;
12.   }
13.  }}
```

Output

```
Enter a number:10
1
3
5
7
9
```

WAP to print factors of a given number

```java
1.  import java.util.*;
2.  class WhileDemo{
3.  public static void main (String[] args){
4.  int i=1,n;
5.  Scanner sc = new Scanner(System.in);
6.  System.out.print("Enter a Number:");
7.  n = sc.nextInt();
8.  System.out.print(" Factors:");
9.  while(i <= n){
10.    if(n%i == 0)
11.      System.out.print(i +",");
12.    i++;
13.  }
14. }}
```

Output

```
Enter a Number:25
Factors:1,5,25
```

Exercise:while

1. WAP to print multiplication table using while loop

2. Write a program that calculates and prints the sum of the even integers from 1 to 10.

for(;;)

Entry Controlled Loop

for

- **for** is an entry controlled loop
- Statements inside the body of **for** are repeatedly executed till the condition is true

```
for (initialization; condition; increment
                /decrement)
{
    // statements
}
```

```
int i = 1;
while (i <= 5) {

System.out.print("Hell
o World!");
    i++;
}
```

```
for(i=1; i <= 5; i++)
{

System.out.print("Hel
lo World!");

}
```

- The **initialization** statement is executed only once, at the beginning of the loop.
- Then, the **condition** is evaluated.
  - If the condition is true, statements inside the body of for loop are executed
  - If the condition is false, the for loop is terminated.
- Then, **increment / decrement** statement is executed
- Again the **condition** is evaluated and so on so forth till the condition is true.

For Loop

- for loop is used to iterate a part of the program several times.

- If the number of iteration is fixed, it is recommended to use for loop.

```java
//code will print 1 to 9
public class ForLoopDemo {
    public static void main(String[] args)
    {
        for(int number=1;number<10;number++)
        {
            System.out.println(number);
        }
    }
}
```

WAP to print odd numbers between 1 to n

```java
1.   import java.util.*;
2.   class MyProgram{
3.   public static void main (String[] args){
4.       int i=1;
5.      Scanner sc = new Scanner(System.in);
6.      n = sc.nextInt();
7.      for(i=1; i<=n; i++)  {
8.          if(i%2==1)
9.              System.out.println(i);
10.     }//for
11.    }//
12.   }
```

WAP to print factors of a given number

```java
1.  import java.util.*;
2.  class MyProgram{
3.  public static void main (String[] args){
4.      int i=1;
5.     Scanner sc = new  Scanner(System.in);
6.    n = sc.nextInt();
7.     for(i=1; i<=n; i++){
8.        if(n%i == 0)
9.           System.out.println(i);
10.   }
11.  }
12. }
```

Exercise: for

 Write a program to print average of n numbers.

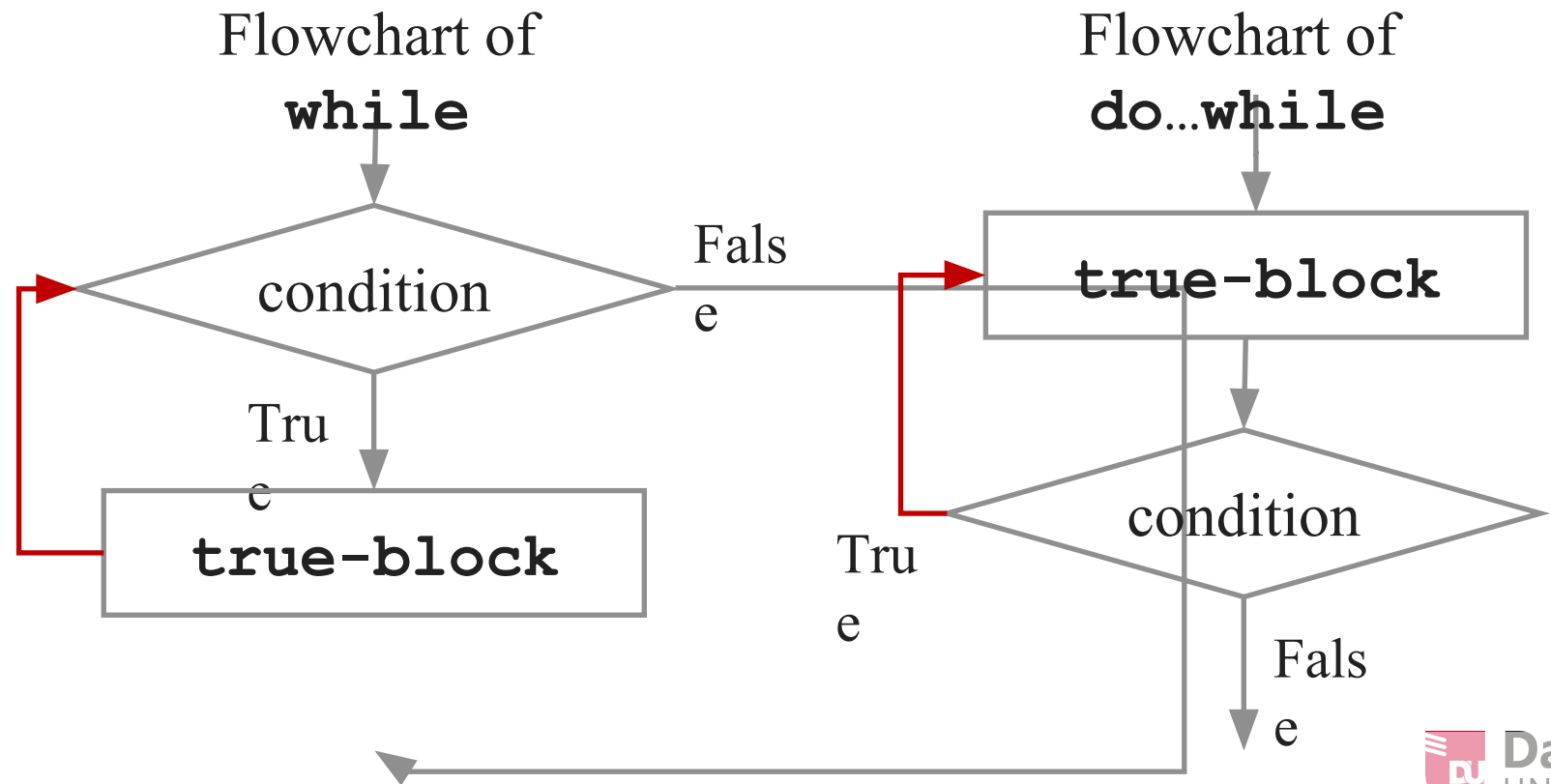 Write a program that calculates and prints the sum of the even integers from 1 to 10.

do…while

Exit Controlled Loop

do…while

- **do…while** is an exit controlled loop.
- Statements inside the body of **do…while** are repeatedly executed till the condition is true.
- **while** loop executes zero or more times, **do…while** loop executes one or more times.

```
do
{
    // true-block
}
while(condition) ;
```

Flowchart of
while

condition

True

Fals
e

true-block

Flowchart of
do...**while**

true-block

condition

True

Fals
e

Darshan
UNIVERSITY

WAP to print 1 to 10 using do-while loop

```java
1.    import java.util.*;
2.    class MyProgram{
3.    public static void main (String[] args){
4.    int i=1;
5.      do{
6.           System.out.println(i);
7.           i++;
8.      }while(i <= 10);
9.    }
10.  }
```

continue

Skip the statement in the iteration

continue

 Sometimes, it is required to skip the remaining statements in the loop and continue with the next iteration.

 `continue` statement is used to skip remaining statements in the loop.

 `continue` is keyword.

WAP to calculate the sum of positive numbers.

```java
1.import java.util.*;
2.class ContinueDemo{
3.public static void main(String[] args) {
4.int a,n,sum=0;
5.Scanner sc = new Scanner(System.in);
6.n = sc.nextInt();
7.for(int i=0;i<n;i++){
8.   a = sc.nextInt();
9.   if(a<0){
10.       continue;
11.       System.out.println("a="+a);//error:unreachable statement
12.   }//if
13.   sum=sum+a;
14.}//for
15. System.out.println("sum="+sum);
16. }
17.}
```
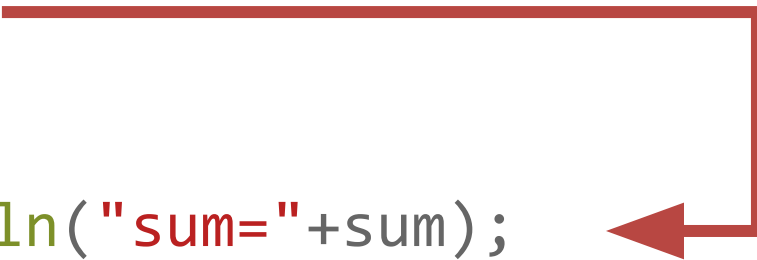
break

Early exit from the loop

break

- Sometimes, it is required to early exit the loop as soon as some situation occurs.

- E.g. searching a particular number in a set of 100 numbers. As soon as the number is found it is desirable to terminate the loop.

- **break** statement is used to jump out of a loop.

- **break** statement provides an early exit from **for**, **while**, **do…while** and **switch** constructs.

- **break** causes exit from the innermost loop or switch.

- **break** is keyword.

WAP to calculate the sum of given numbers. User will enter -1 to terminate.

```java
1.import java.util.*;
2.class BreakDemo{
3.public static void main (String[] args){
4.      int a,sum=0;
5.      System.out.println("enter numbers_ enter -1 to break");
6.      Scanner sc = new Scanner(System.in);
7.      while(true){
8.          a = sc.nextInt();
9.          if(a==-1)
10.             break;
11.         sum=sum+a;
12.     }//while
13.     System.out.println("sum="+sum);
14. }
15.}
```
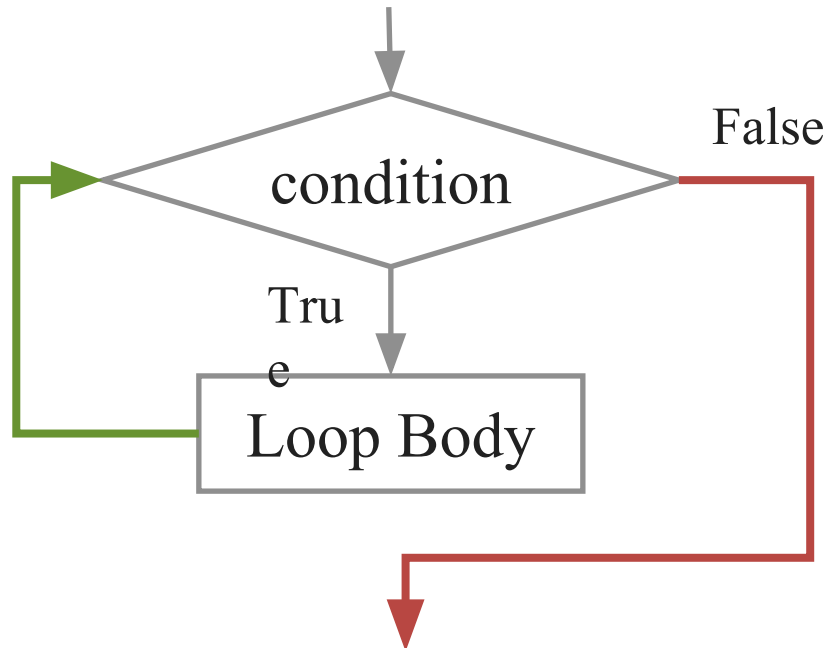
Darshan UNIVERSITY

Types of loops

Entry Control Loop	Entry Control Loop	Exit Control Loop	Virtual Loop

```
int i=1;
while(i<=10)
{
    i++;
}
```

```
int i;
for(i=1;i<=10;i++)
{
    i++;
}
```

```
int i=1;
do
{
        i++;
}
while(i<=10);
```
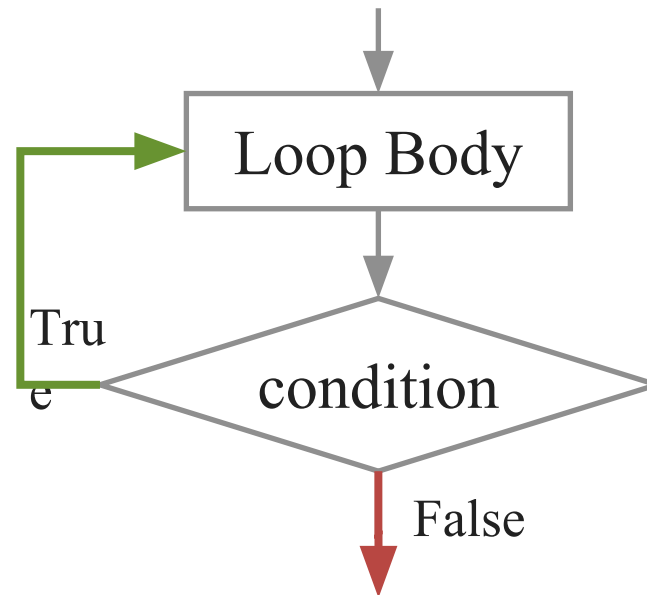
```
    int i=1;
p: i++;
    if(i<=10)
        goto p;
```
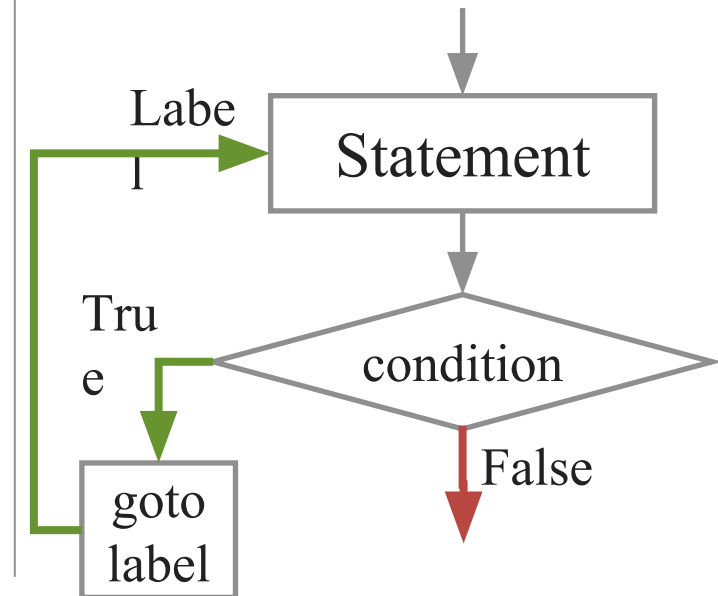
nested loop

loop within a loop

WAP to print given pattern (nested loop)

```
*
**
***
****
*****
```

```java
1. class PatternDemo{
2. public static void main(String[] args) {
3.     int n=5;
4.     for(int i=1;i<=n;i++){
5.         for(int j=1;j<=i;j++){
6.             System.out.print("*");
7.         }//for j
8.         System.out.println();
9.     }//outer for i
10.    }
11. }
```

WAP to print given pattern (nested loop)

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

```java
1. class PatternDemo{
2. public static void main(String[] args) {
3.    int n=5;
4.    for(int i=1;i<=n;i++){
5.        for(int j=1;j<=i;j++){
6.                System.out.print(j+"\t");
7.        }//for j
8.            System.out.println();
9.    }//outer for i
10.    }
11. }
```

Programs to perform (Looping Statements)

- Write a program to print first n odd numbers.
- Write a program to check that the given number is prime or not.
- Write a program to draw given patterns,

```
* * * * *        *              * * * * *            *              *              *          * * * * *          1
* * * * *      * *              * * * *            * *            * *            ***          * * * *            2 3
* * * * *      * * *            * * *            * * *          * * *          *****          * * *              4 5 6
* * * * *      * * * *          * *            * * * *        * * * *        *******          * *                7 8 9 10
* * * * *      * * * * *        *            * * * * *      * * * * *      *********          *                  11 12 13 14 15
                                                                         *******            * *
                                                                         *****              * * *
                                                                         ***                * * * *
                                                                         *                  * * * * *
```
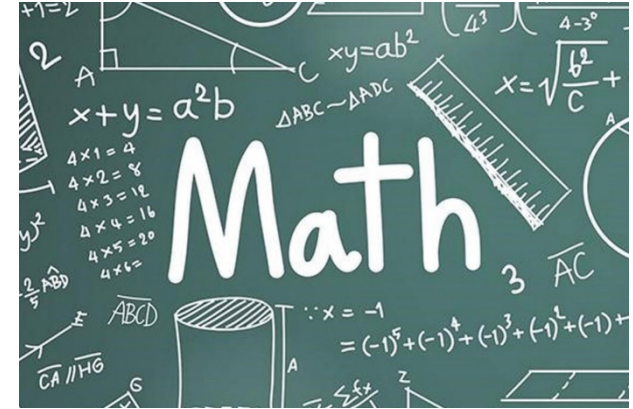
```
        *                            *
      * *                          * *
    *   *                        *   *
  *     *                      *     *
*********                    *       *
                               *   *
                                 * *
                                   *
```

Mathematical functions

Math class

- The Java Math class provides more **advanced mathematical calculations** other than arithmetic operator.

- The **java.lang.Math** class contains methods which performs basic numeric operations such as the **elementary exponential**, **logarithm**, **square root**, and **trigonometric** functions.

- All the **methods** of class Math are **static**.

- **Fields :**
  - Math class comes with two important **static** fields
    - **E** : returns double value of **Euler's number** (i.e 2.718281828459045).
    - **PI** : returns double value of **PI** (i.e. 3.141592653589793).

Methods of class Math

Method	description	Example

Methods of class Math (Cont.)

Method	description	Example

Methods of class Math (Cont.)

Method	description	Example

Methods of class Math (Cont.)

Method	description	Example

Methods of class Math (Cont.)

Method	description	Example

Math Example

```java
public class MathDemo {
    public static void main(String[] args) {
        double sinValue = Math.sin(Math.PI / 2);
        double cosValue = Math.cos(Math.toRadians(80));
        int randomNumber = (int)(Math.random() * 100);
        // values in Math class must be given in Radians
        // (not in degree)
        System.out.println("sin(90) = " + sinValue);
        System.out.println("cos(80) = " + cosValue);
        System.out.println("Random = " + randomNumber);
    }
}
```

```
sin(90) = 1.0
cos(80) = 0.17364817766693041
Random = 29
```

Thank You