DAILY ASSESSMENT

Date:	**9-6-2020**	**Name:**	**Kavyashree m**
Course:	**VLSI**	**USN:**	**4al15ec036**
Topic:	**MOSFET - Enhancement Type MOSFET Explained, GATE 2009 and 20121 ECE operating region and output voltage of CMOS inverter , MOSFET vth based problems, MOSFET problems and solutions, TRICK to implement 4:1 mux using TRANSMISSION GATE &amp, Realization of logic function using Multiplexer**	**Semester & Section:**	**8th A**
Github Repository:	**kavya**		

FORENOON SESSION DETAILS

Image of session

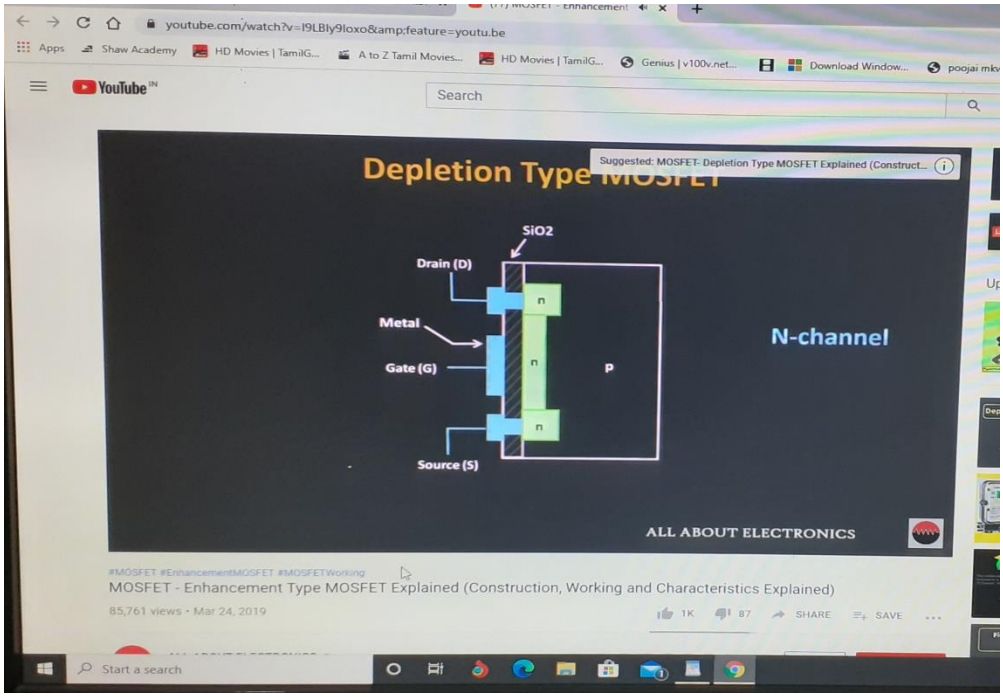


Fig 1: Enhancement Type MOSFET Explained

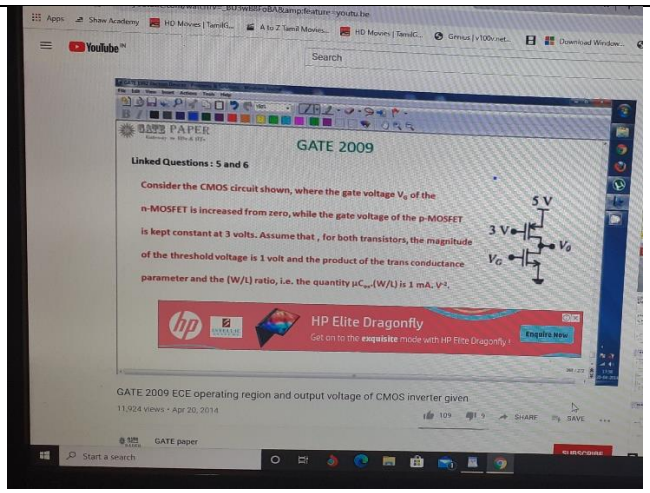


Fig 2: **GATE 2009 and 20121 ECE operating region and output voltage**

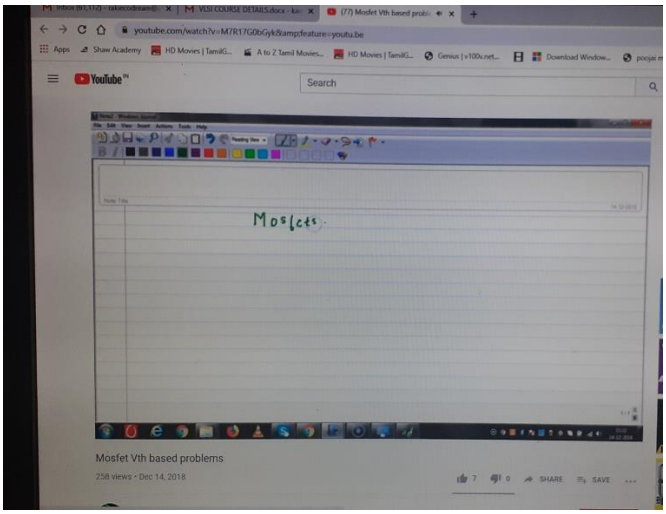


Fig 3: MOSFET vth based problems

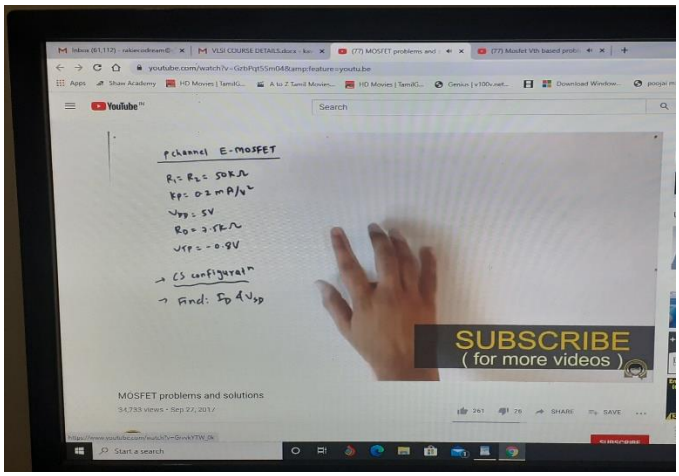


Fig 4: MOSFET problems and solutions

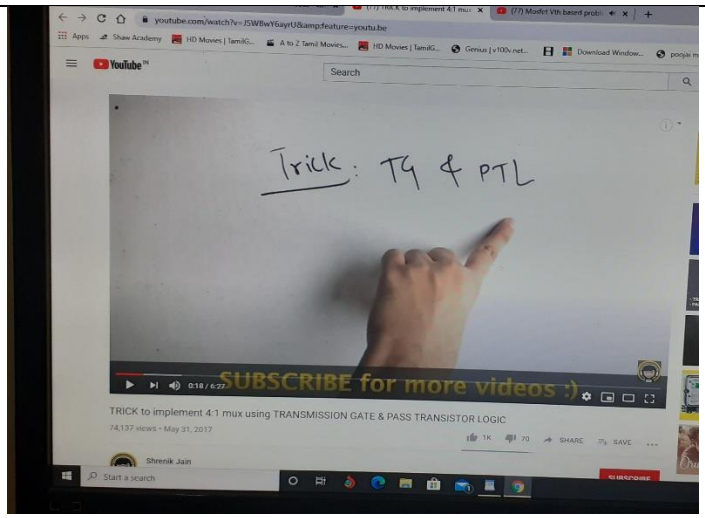


Fig 5: TRICK to implement 4:1 mux using TRANSMISSION GATE & PASS TRANSISTOR LOGIC

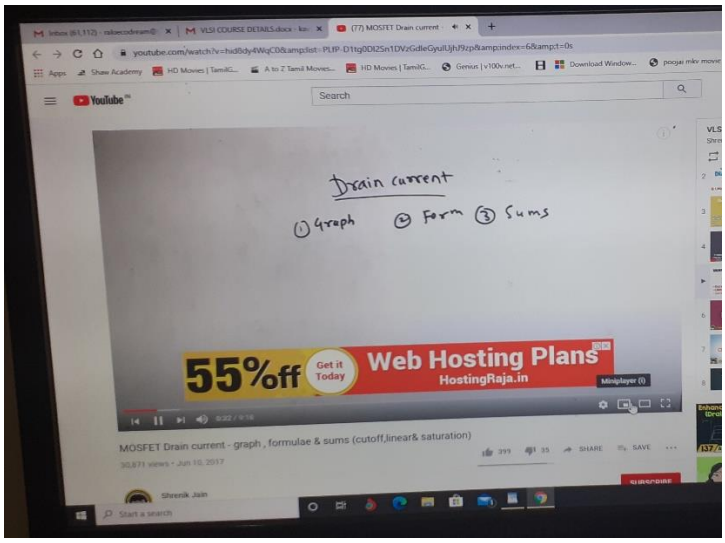


Fig 6: MOSFET Drain current

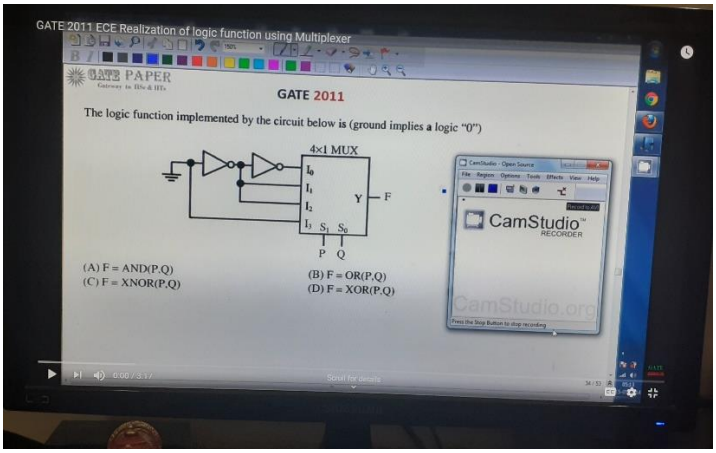
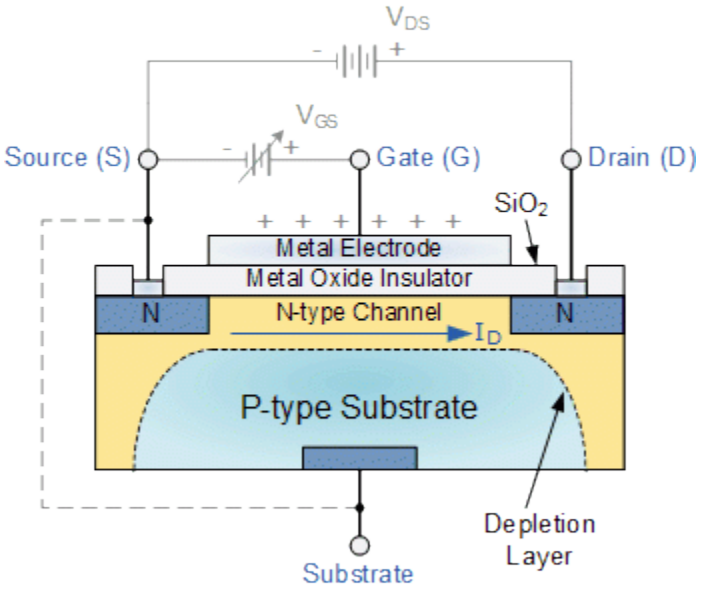


Fig 7: Realization of logic function using Multiplexer

MOSFET

Enhancement Type MOSFET Explained



The IGFET or MOSFET is a voltage controlled field effect transistor that differs from a JFET in that it has a “Metal Oxide” Gate electrode which is electrically insulated from the main semiconductor n-channel or p-channel by a very thin layer of insulating material usually silicon dioxide, commonly known as glass.

This ultra thin insulated metal gate electrode can be thought of as one plate of a capacitor. The isolation of the controlling Gate makes the input resistance of the MOSFET extremely high way up in the Mega-ohms (MΩ) region thereby making it almost infinite.

As the Gate terminal is electrically isolated from the main current carrying channel between the drain and source, “NO current flows into the gate” and just like the JFET, the MOSFET also acts like a voltage controlled resistor where the current flowing through the main channel between the Drain and Source is proportional to the input voltage. the MOSFETs very high input resistance can easily accumulate large amounts

of static charge resulting in the MOSFET becoming easily damaged unless carefully handled or protected.

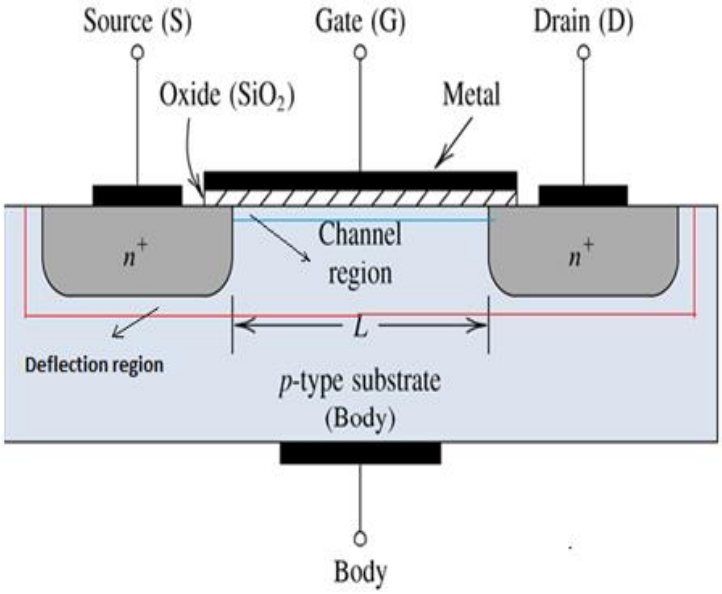
Like the previous JFET tutorial, MOSFETs are three terminal devices with a Gate, Drain and Source and both P-channel (PMOS) and N-channel (NMOS) MOSFETs are available. The main difference this time is that MOSFETs are available in two basic forms:

- Depletion Type – the transistor requires the Gate-Source voltage, (V_{GS}) to switch the device "OFF". The depletion mode MOSFET is equivalent to a "Normally Closed" switch.
- Enhancement Type – the transistor requires a Gate-Source voltage, (V_{GS}) to switch the device "ON". The enhancement mode MOSFET is equivalent to a "Normally Open" switch.

Working Principle of MOSFET

The aim of the MOSFET is to be able to control the voltage and current flow between the source and drain. It works almost as a switch. The working of MOSFET depends upon the MOS capacitor. The MOS capacitor is the main part of MOSFET. The semiconductor surface at the below oxide layer which is located between source and drain terminal. It can be inverted from p-type to n-type by applying a positive or negative gate voltages respectively. When we apply the positive gate voltage the holes present under the oxide layer with a repulsive force and holes are pushed downward with the substrate. The depletion region populated by the bound negative charges which are associated with the acceptor atoms. The electrons reach channel is formed. The positive voltage also attracts electrons from the n+ source and drain regions into the channel. Now, if a voltage is applied between the drain and source, the current flows freely between the source and drain and the gate voltage controls the electrons in the channel.

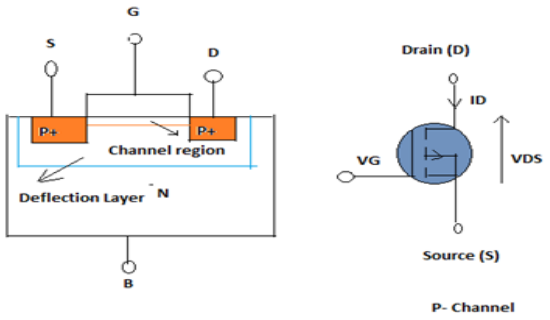
Instead of positive voltage if we apply negative voltage , a hole channel will be formed under the oxide layer.



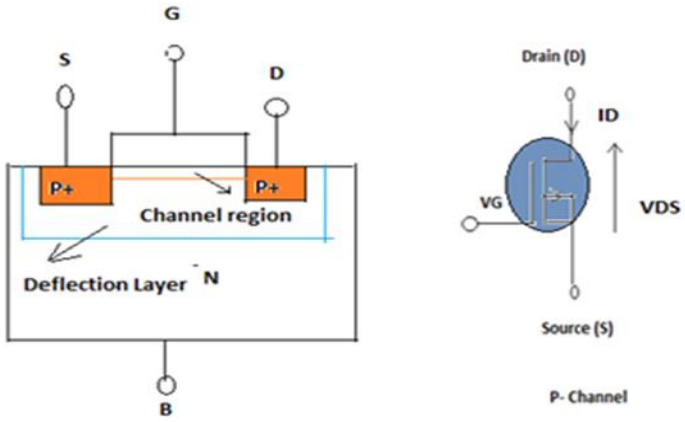
MOSFET Block Diagram

P-Channel MOSFET:

The P- Channel MOSFET has a P- Channel region between source and drain. It is a four terminal device such as gate, drain, source, body. The drain and source are heavily doped p+ region and the body or substrate is n-type. The flow of current is positively charged holes. When we apply the negative gate voltage, the electrons present under the oxide layer with are pushed downward into the substrate with a repulsive force. The depletion region populated by the bound positive charges which are associated with the donor atoms. The negative gate voltage also attracts holes from p+ source and drain region into the channel region.



Enhanced mode

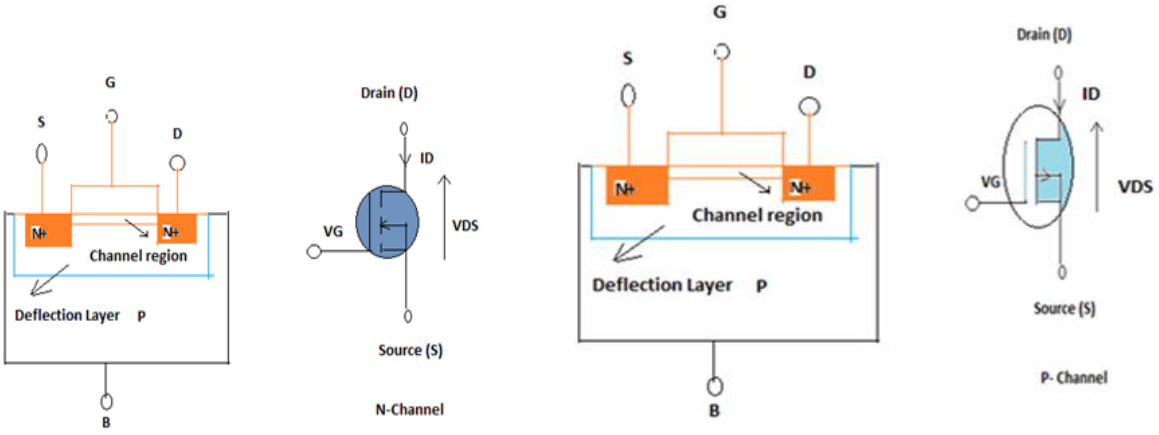


Depletion Mode

N- Channel MOSFET:

The N-Channel MOSFET has a N- channel region between source and drain It is a four terminal device such as gate, drain , source , body. This type of MOSFET the drain and source are heavily doped n+ region and the substrate or body is P- type. The current flows due to the negatively charged electrons. When we apply the positive gate voltage the holes present under the oxide layer pushed downward into the substrate with a repulsive force. The depletion region is populated by the bound negative charges which are associated with the acceptor atoms. The electrons reach channel is formed. The positive

voltage also attracts electrons from the n+ source and drain regions into the channel. Now, if a voltage is applied between the drain and source the current flows freely between the source and drain and the gate voltage controls the electrons in the channel. Instead of positive voltage if we apply negative voltage a hole channel will be formed under the oxide layer.



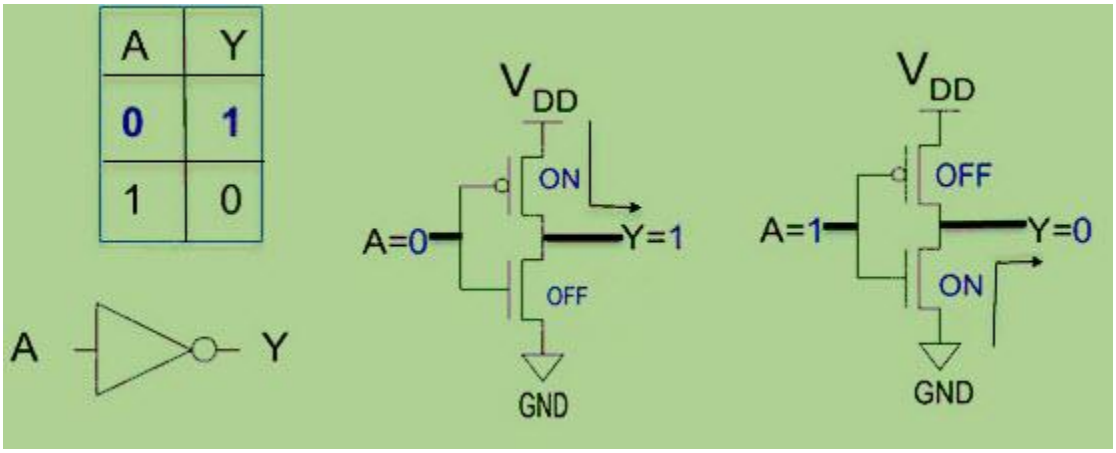
Enhanced mode Depletion Mode

GATE 2009 and 20121 ECE operating region and output voltage of CMOS inverter

Complementary metal–oxide–semiconductor (CMOS),also known as complementary-symmetry metal–oxide–semiconductor (COS-MOS), is a type of metal–oxide–semiconductor field-effect transistor (MOSFET) fabrication process that uses complementary and symmetrical pairs of p-type and n-type MOSFETs for logic functions.

CMOS Inverter

The inverter circuit as shown in the figure below. It consists of PMOS and NMOS FET. The input A serves as the gate voltage for both transistors.



CMOS Inverter

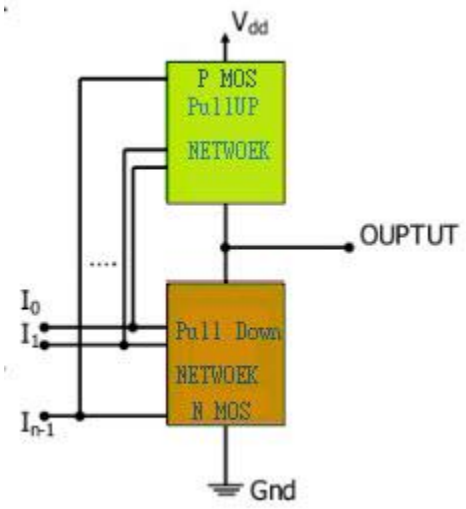
The NMOS transistor has an input from Vss (ground) and PMOS transistor has an input from Vdd. The terminal Y is output. When a high voltage (~ Vdd) is given at input terminal (A) of the inverter, the PMOS becomes open circuit and NMOS switched OFF so the output will be pulled down to Vss.

Working Principle

In CMOS technology, both N-type and P-type transistors are used to design logic functions. The same signal which turns ON a transistor of one type is used to turn OFF a transistor of the other type. This characteristic allows the design of logic devices using only simple switches, without the need for a pull-up resistor.

In CMOS logic gates a collection of n-type MOSFETs is arranged in a pull-down network between the output and the low voltage power supply rail (Vss or quite often ground). Instead of the load resistor of NMOS logic gates, CMOS logic gates have a collection of p-type MOSFETs in a pull-up network between the output and the higher-voltage rail .

Thus, if both a p-type and n-type transistor have their gates connected to the same input, the p-type MOSFET will be ON when the n-type MOSFET is OFF, and vice-versa. The networks are arranged such that one is ON and the other OFF for any input pattern as shown in the figure below.

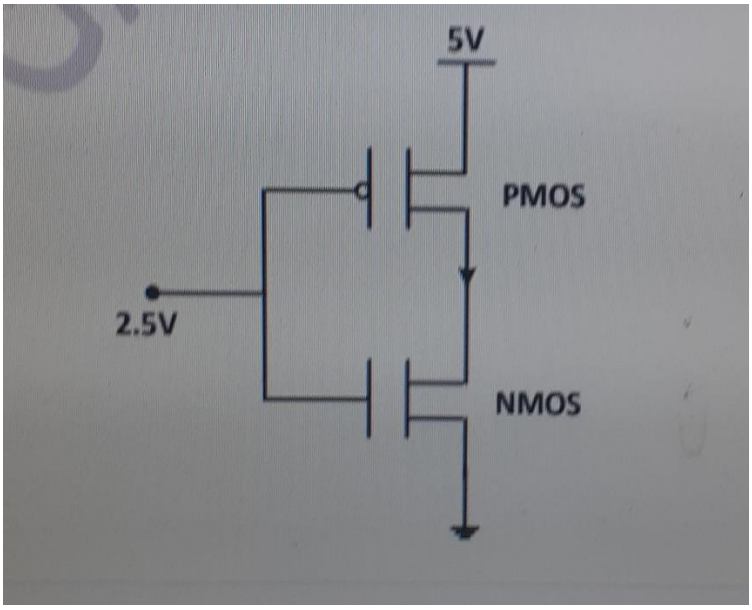


CMOS Logic Gate using Pull-Up and Pull-Down Networks

CMOS offers relatively high speed, low power dissipation, high noise margins in both states, and will operate over a wide range of source and input voltages.

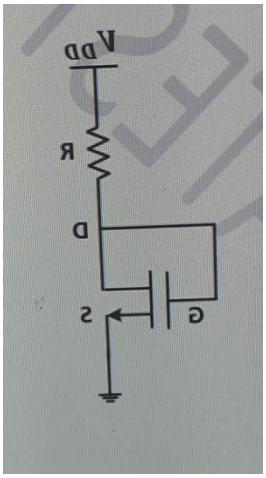
MOSFET problems and solutions

1)In the C MOS inverter circuit shown if the transconductance parameters of N MOS and P MOS transistor are $Kn = Kp = \mu nCOX\ Wn\ Ln = \mu pCOX\ Wp\ Lp = 40\ \mu A\ V\ 2/$ and threshold voltages are $VT = 1V$ the current I is



Sol) Given $\boldsymbol{Kn} = \boldsymbol{Kp} = \mathbf{40}\ \boldsymbol{\mu A}\ \boldsymbol{V}\ \mathbf{2}/\boldsymbol{VT} = \mathbf{1}\ \boldsymbol{V}$ The device is in saturation. So the current is given by $\boldsymbol{IDS} = \boldsymbol{Kn}\ \mathbf{2}\ (\boldsymbol{VGS} - \boldsymbol{VT}\ )\ \mathbf{2}\ \mathbf{40}\ \mathbf{2}\ (\mathbf{2}.\ \mathbf{5} - \mathbf{1})\ \mathbf{2} = \mathbf{20} \times (\mathbf{1}.\ \mathbf{5})\ \mathbf{2} = \mathbf{45}\ \boldsymbol{\mu A}$.

2) For the n – channel MOS transistor shown in figure, the threshold voltage VTH is 0.8V. Neglect channel length modulation effects. When the drain voltage VD = 1.6, the drain current ID was found to be 0.5 mA. If VD is adjusted to be 2V by changing the values of R and VDD, the new value of ID (in m A) is



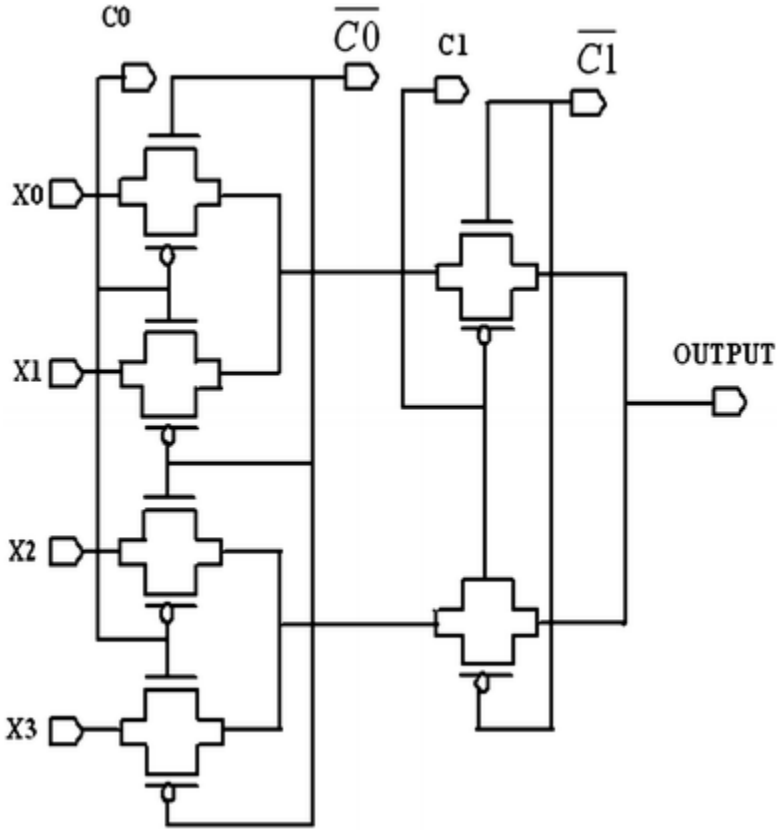
Sol) Given, $\boldsymbol{VTH} = \mathbf{0}.\ \mathbf{8}\ \boldsymbol{V}\ \boldsymbol{VD} = \mathbf{1}.\ \mathbf{6}\ \boldsymbol{V}\ \boldsymbol{ID} = \mathbf{0}.\ \mathbf{5}\ \boldsymbol{mA}$ For the given figure we notice that Gate is connected to drain So, $\boldsymbol{VGS} = \boldsymbol{VDS} = \boldsymbol{VD}$ In saturation ID is given by $\boldsymbol{IDS} = \boldsymbol{K}(\boldsymbol{VGS} - \boldsymbol{VT}\ )\ \mathbf{2}\ \boldsymbol{So}, \boldsymbol{ID2}\ \boldsymbol{ID1} = [\boldsymbol{VGS2} - \boldsymbol{VT}]\ \mathbf{2}\ [\boldsymbol{VGS1} - \boldsymbol{VT}]\ \mathbf{2}\ \boldsymbol{So}, \boldsymbol{ID2}\ \boldsymbol{ID1} = (\mathbf{2} - \mathbf{0}.\ \mathbf{8})\ \mathbf{2}\ (\mathbf{1}.\ \mathbf{6} - \mathbf{0}.\ \mathbf{8})\ \mathbf{2} = \mathbf{2}.\ \mathbf{25}\ \boldsymbol{or}, \boldsymbol{ID2} = \mathbf{2}.\ \mathbf{25} \times \mathbf{0}.\ \mathbf{5} = \mathbf{1}.\ \mathbf{125}\ \boldsymbol{m}$.

TRICK to implement 4:1 mux using TRANSMISSION GATE &amp

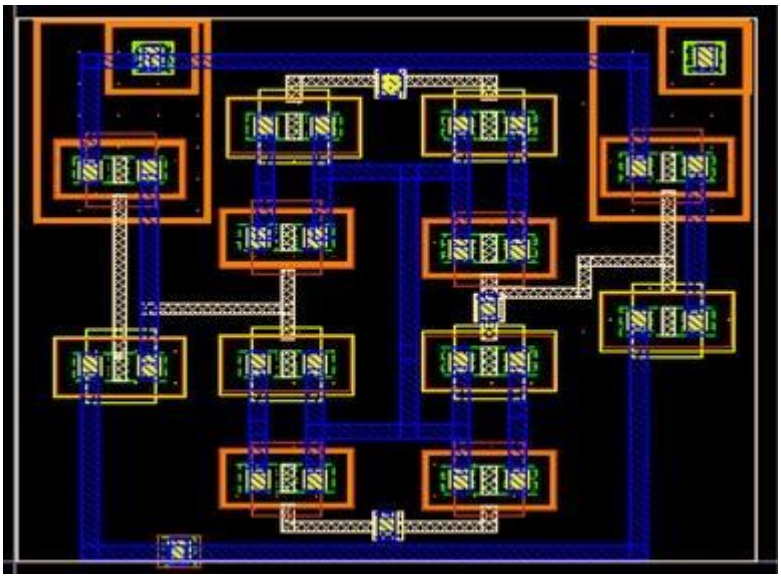
Transmission gate logic based 4:1 MUX

This design is the transmission gate type of MUX structure implemented with very minimum transistors compared to the conventional CMOS based design. The back-to-back connected PMOS and NMOS arrangement acts as a switch is so called transmission gate. NMOS devices pass a strong 0 but a weak 1, while PMOS pass a strong 1 but a

weak 0. The transmission gate combines the best of both the properties by placing NMOS in parallel with the PMOS device. Four transmission gates are connected as to form a MUX structure.



The advances in the CMOS processes are generally complex and somewhat inhibit the visualization of all the mask levels that are used in the actual fabrication process. Nevertheless, the design process can be abstracted to a manageable number of conceptual layout levels that represent the physical features observed in the final silicon wafer. An advantage of the new MUX design is the remarkable gain in terms of transistors count. To the best of our knowledge, no 4:1 MUX has been realized with so few devices. Hence, the gain in area is a central result for the proposed MUX.



Power and current consumption

Digital CMOS circuit may have three major sources of power dissipation namely dynamic, short and leakage power. Hence, the total power consumed by every MUX style can be evaluatedusing the below equation

Ptot=Pdyn+Psc+Pleak=CL·Vdd·V·fclk+ISC·Vdd+Ileak·VddPtot=Pdyn+Psc+Pleak=CL·Vdd·V·fclk+ISC·Vdd+Ileak·Vdd

Thus, for low-power design, the important task is to minimize $CL \cdot V_{dd} \cdot V \cdot f_{Clk}$ while retaining required functionality. The first term P_{dyn} represents the switching component of power, the next component P_{sc} is the short circuit power and P_{leak} is the leakage power. Where, CL is the loading capacitance, f_{Clk} is the clock frequency which is actually the probability at which Logic 0 to 1 transition occurs . V_{dd} is the supply voltage, V is the output voltage swing which is equal to V_{dd}; but, in some logic circuits, such as proposed transmission logic implementations, the voltage swing on some internal nodes may be slightly less.

Realization of logic function using Multiplexer

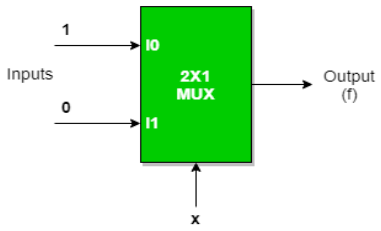
Multiplexers in Digital Logic

It is a combinational circuit which have many data inputs and single output depending on control or select inputs. For N input lines, log n (base2) selection lines, or we can say that for 2^n input lines, n selection lines are required. Multiplexers are also known as **"Data n selector, parallel to serial convertor, many to one circuit, universal logic circuit"**. Multiplexers are mainly used to increase amount of the data that can be sent over the network within certain amount of time and bandwidth.

Multiplexer can act as universal combinational circuit. All the standard logic gates can be implemented with multiplexers.

a) Implementation of NOT gate using 2 : 1 Mux

NOT Gate :

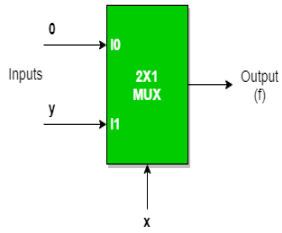


Truth Table

x	f
0	1
1	0

b) Implementation of AND gate using 2 : 1 Mux

AND GATE:



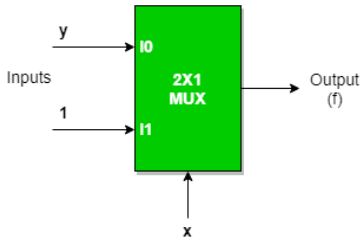
Truth Table

x	y	f	f→y
0	0	0	f = 0
0	1	0	
1	0	0	f = y
1	1	1	

This implementation is done using "n-1" selection lines.

c) Implementation of OR gate using 2 : 1 Mux using "n-1" selection lines.

OR GATE



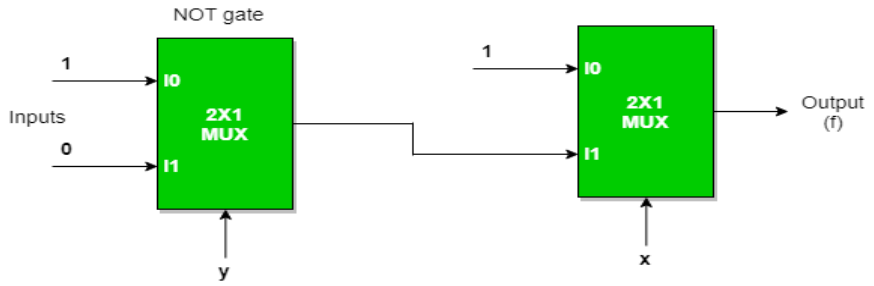
Truth Table

x	y	f	f→y
0	0	0	f = y
0	1	1	
1	0	1	f = 1
1	1	1	

Implementation of NAND, NOR, XOR and XNOR gates requires two 2:1 Mux. First multiplexer will act as NOT gate which will provide complemented input to the second multiplexer.

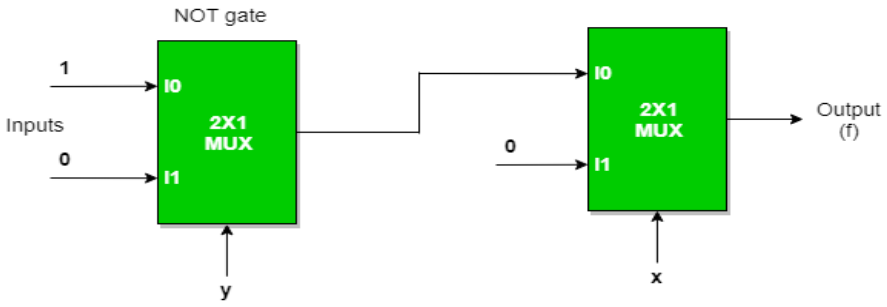
d) Implementation of NAND gate using 2 : 1 Mux

NAND GATE



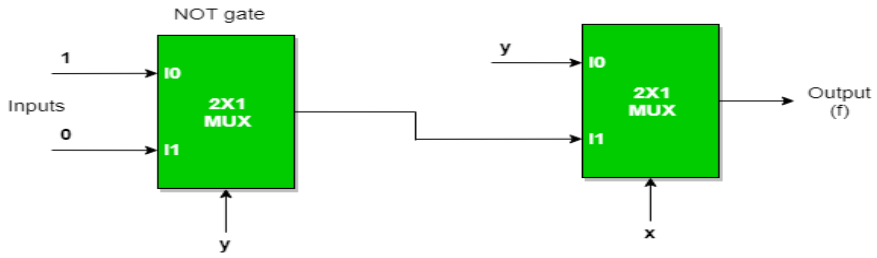
e) Implementation of NOR gate using 2 : 1 Mux

NOR GATE



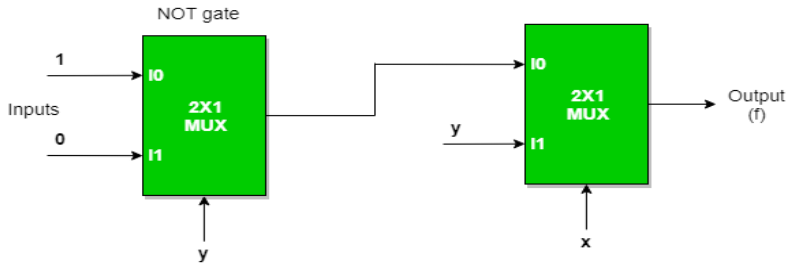
f) Implementation of EX-OR gate using 2 : 1 Mux

EX-OR GATE



g) Implementation of EX-NOR gate using 2 : 1 Mux

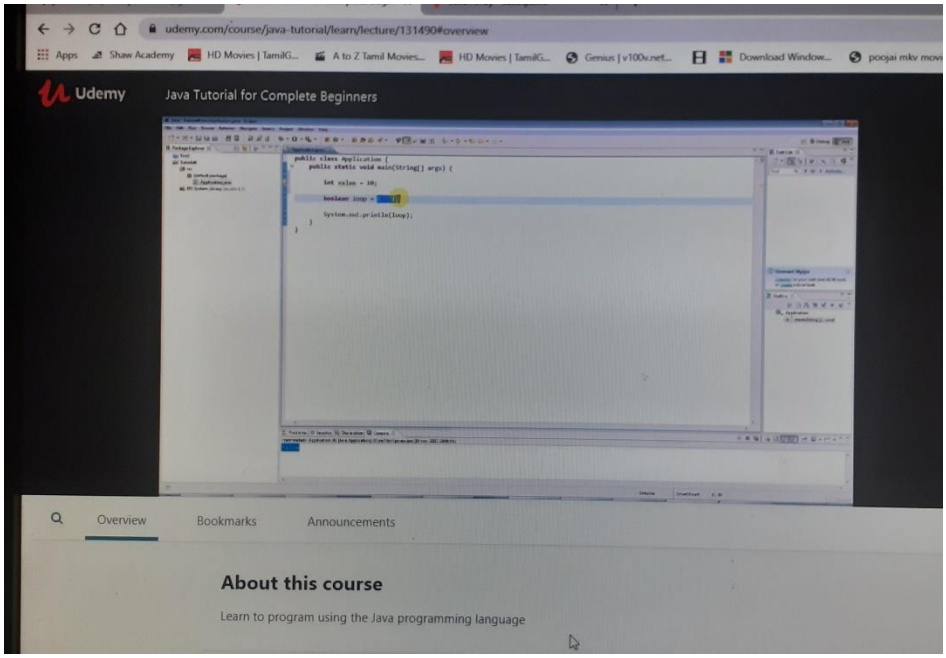
EX-NOR GATE



AFTERNOON SESSION DETAILS

Date:	**9-6-2020**	**Name:**	**Kavyashree m**
Course:	**Java**	**USN:**	**4al15ec036**
Topic:	**Hello world program ,using varaibles,strings,While loop,for loop,if, do while, switches ,array**	**Semester & Section:**	**8th A**
Github Repository:	**kavya**		

Image of session



Java

Hello world Java program

```
// a small Java program
public class HelloWorld {
    public static void main(String[] args) {
```

```
    System.out.println("Hello World");
  }
```

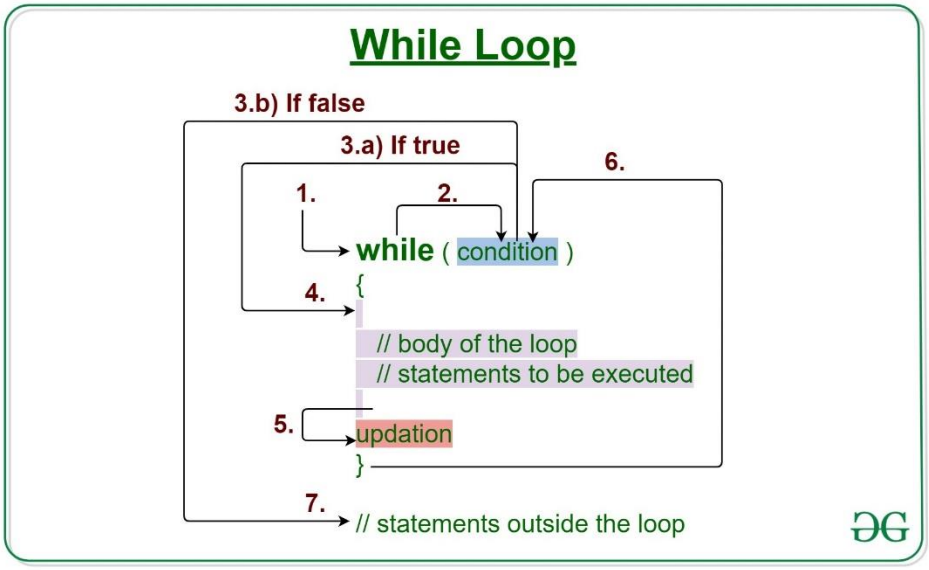
Using variable

A variable is a name given to a memory location. It is the basic unit of storage in a program.

- The value stored in a variable can be changed during program execution.
- A variable is only a name given to a memory location, all the operations done on the variable effects that memory location.
- In Java, all the variables must be declared before use**.**

While loop

Java while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.



Syntax:

```
while (test_expression)
{
```

```
  // statements


 update_expression;
}
```

The various parts of the While loop are:

1. Test Expression: In this expression we have to test the condition. If the condition evaluates to true then we will execute the body of the loop and go to update expression. Otherwise, we will exit from the while loop.

   Example:

```
i <= 10
```

2. Update Expression: After executing the loop body, this expression increments/decrements the loop variable by some value.

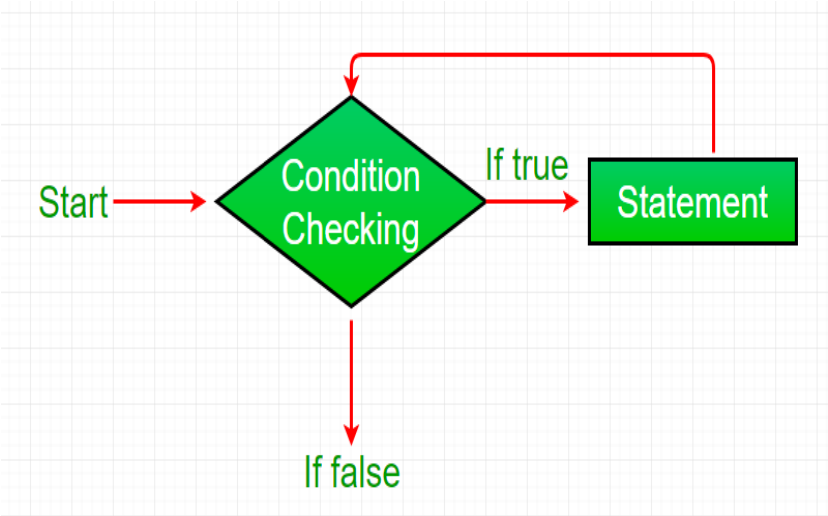
   Example:

```
i++;
```

How does a While loop executes?

1. Control falls into the while loop.
2. The flow jumps to Condition
3. Condition is tested.
    a. If Condition yields true, the flow goes into the Body.
    b. If Condition yields false, the flow goes outside the loop
4. The statements inside the body of the loop get executed.
5. Updation takes place.
6. Control flows back to Step 2.
7. The do-while loop has ended and the flow has gone outside.

Flow chart while loop (for Control Flow):



Example 1: This program will try to print “Hello World” 5 times.

filter_none

edit

play_arrow

brightness_4

```
// Java program to illustrate while loop.

class whileLoopDemo {
    public static void main(String args[])
    {
        // initialization expression
        int i = 1;

        // test expression
        while (i < 6) {
            System.out.println("Hello World");
```

```
            // update expression
            i++;
        }
    }
}
```

Output:

```
Hello World
Hello World
Hello World
Hello World
Hello World
```

For loop in java

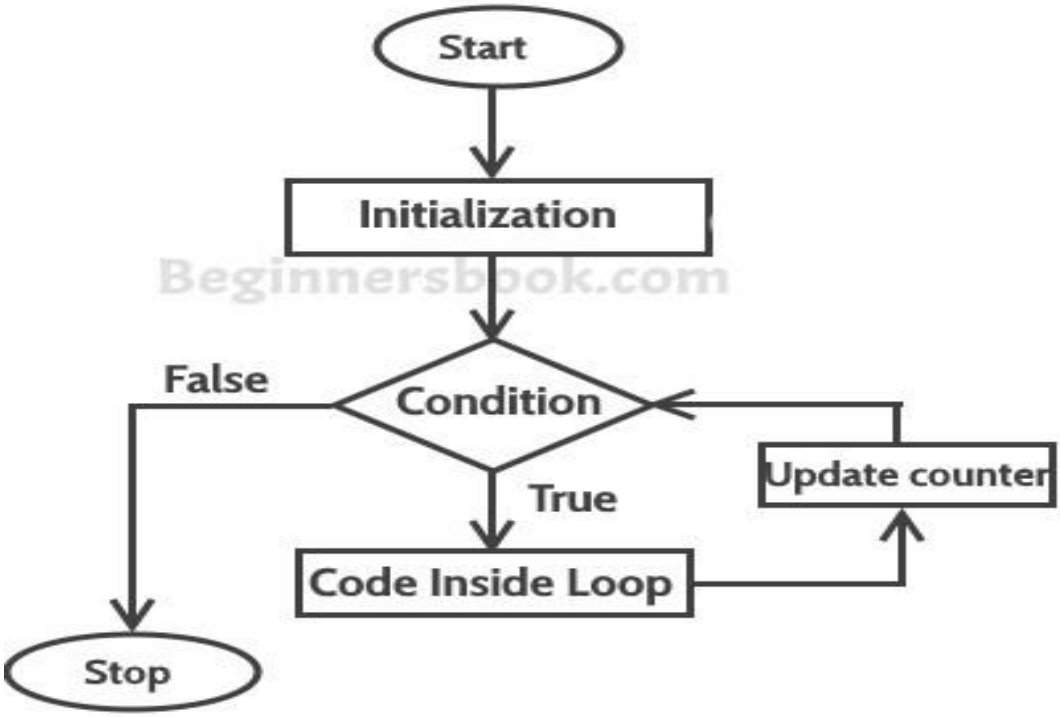
Loops are used to execute a set of statements repeatedly until a particular condition is satisfied. In Java we have three types of basic loops: for, while and do-while. In this tutorial we will learn how to use “for loop” in Java.

Syntax of for loop:

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

Flow of Execution of the for Loop

As a program executes, the interpreter always keeps track of which statement is about to be executed. We call this the control flow, or the flow of execution of the program.



If statement in java

Java if Statement

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

      int number = 10;

      // checks if number is greater than 0
      if (number > 0) {
         System.out.println("The number is positive.");
      }
      System.out.println("This statement is always executed.");
   }
}
```

Output:

The number is positive.

This statement is always executed.

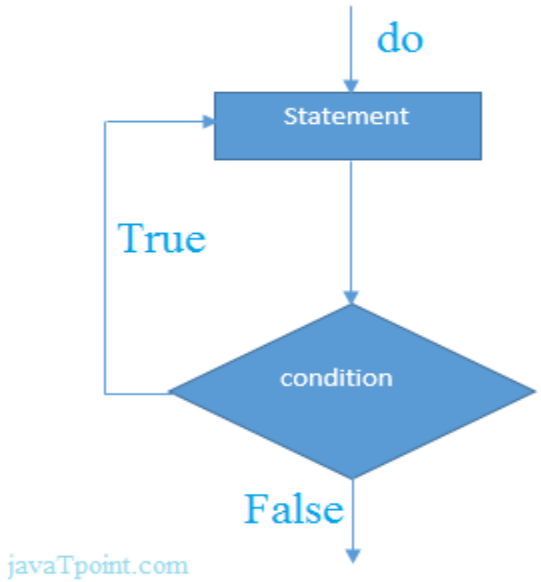
do-while Loop

The do-while loop is used to iterate a part of the program several times. If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use do-while loop.

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

Syntax:

1. **do**{
2. //code to be executed
3. }**while**(condition);



Example:

1. **public class** DoWhileExample {
2. **public static void** main(String[] args) {

```
3.  int i=1;
4.  do{
5.      System.out.println(i);
6.  i++;
7.  }while(i<=10);
8.  }
9.  }
```

Output:

```
1
2
3
4
5
6
7
8
9
10
```

Strings

Strings in Java are Objects that are backed internally by a char array. Since arrays are immutable(cannot grow), Strings are immutable as well. Whenever a change to a String is made, an entirely new String is created.

Below is the basic syntax for declaring a string in Java programming language.

Syntax:

<String_Type> <string_variable> = "<sequence_of_string>";

An Example that shows how to declare String

filter_none

edit

play_arrow

brightness_4

```
// Java code to illustrate String

import java.io.*;
import java.lang.*;

class Test {
    public static void main(String[] args)
    {
        // Declare String without using new operator
        String s = "GeeksforGeeks";

        // Prints the String.
        System.out.println("String s = " + s);

        // Declare String using new operator
        String s1 = new String("GeeksforGeeks");

        // Prints the String.
        System.out.println("String s1 = " + s1);
    }
}
```

Output:

String s = GeeksforGeeks

String s1 = GeeksforGeeks

Switches

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.

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
}
```

Example:

```
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);
  }
}
```

Output

```
Well done
Your grade is C
```

Arrays

Java array is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.Array in Java is index-based, the first element of the array is stored at the 0th index, 2nd element is stored on 1st index and so on.

In Java, array is an object of a dynamically generated class. Java array inherits the Object class, and implements the Serializable as well as Cloneable interfaces. We can store primitive values or objects in an array in Java. Like C/C++, we can also create single dimentional or multidimentional arrays in Java.