**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **12/06/2020** | **Name:** | **Nayanashree K S** |
| **Course:** | **VLSI design** | **USN:** | **4AL16EC042** |
| **Topic:** | **MOS transistor basics** | **Semester & Section:** | **8 A** |
| **Github Repository:** | **nayana\_online** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report** Construction of a MOSFET The construction of a MOSFET is a bit similar to the FET. An oxide layer is deposited on the substrate to which the gate terminal is connected. This oxide layer acts as an insulator (sio2 insulates from the substrate), and hence the MOSFET has another name as IGFET. In the construction of MOSFET, a lightly doped substrate, is diffused with a heavily doped region. Depending upon the substrate used, they are called as **P-type** and **N-type** MOSFETs.  The following figure shows the construction of a MOSFET.  MOSFET Construction  The voltage at gate controls the operation of the MOSFET. In this case, both positive and negative voltages can be applied on the gate as it is insulated from the channel. With negative gate bias voltage, it acts as **depletion MOSFET** while with positive gate bias voltage it acts as an **Enhancement MOSFET**.  The **N-channel MOSFETs** are simply called as **NMOS**. The symbols for N-channel MOSFET are as given below.  N-Channel MOSFET  The **P-channel MOSFETs** are simply called as **PMOS**. The symbols for P-channel MOSFET are as given below.  P-channel MOSFET  Now, let us go through the constructional details of an N-channel MOSFET. Usually an NChannel MOSFET is considered for explanation as this one is mostly used. Also, there is no need to mention that the study of one type explains the other too. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **12/6/2020** | **Name:** | **Nayanashree K S** | |
| **Course:** | **Java** | **USN:** | **4al16ec042** | |
| **Topic:** | **Generics and Wildcards, Anonymous Classes, Reading Files Using Scanner, Handling Exceptions, Multiple Exceptions….** | **Semester & Section:** | **8 A** | |
| **AFTERNOON SESSION DETAILS** | | | |
| 3  **REPORT** **Wildcards in Java** The question mark (?) is known as the wildcard in generic programming . It represents an unknown type. The wildcard can be used in a variety of situations such as the type of a parameter, field, or local variable; sometimes as a return type. Unlike arrays, different instantiations of a generic type are not compatible with each other, not even explicitly. This incompatibility may be softened by the wildcard if ? is used as an actual type parameter.  **Types of wildcards in Java:**   1. **Upper Bounded Wildcards:**These wildcards can be used when you want to relax the restrictions on a variable. For example, say you want to write a method that works on List < integer >, List < double >, and List < number > , you can do this  using an upper bounded wildcard. To declare an upper-bounded wildcard, use the wildcard character (‘?’), followed by the extends keyword, followed by its upper bound.   //Java program to demonstrate Upper Bounded Wildcards  import java.util.Arrays;  import java.util.List;  class WildcardDemo  {  public static void main(String[] args)  {    //Upper Bounded Integer List  List<Integer> list1= Arrays.asList(4,5,6,7);    //printing the sum of elements in list  System.out.println("Total sum is:"+sum(list1));  //Double list  List<Double> list2=Arrays.asList(4.1,5.1,6.1);    //printing the sum of elements in list  System.out.print("Total sum is:"+sum(list2));  }  private static double sum(List<? extends Number> list)  {  double sum=0.0;  for (Number i: list)  {  sum+=i.doubleValue();  }  return sum;  }  } **Abstract Classes in Java** In C++, if a class has at least one pure virtual function, then the class becomes abstract. Unlike C++, in Java, a separate keyword abstract is used to make a class abstract.  // An example abstract class in Java  abstract class Shape {  int color;  // An abstract function (like a pure virtual function in C++)  abstract void draw();  } | | | |