1. public class MatrixAddition {  
    public static void main(String[] args) {  
    int m1[][]={{1,2,3},  
    {1,2,4}};  
    int m2[][]={{1,2,5},  
    {6,7,8}};  
   // int sum[][]={{0,0,0},{0,0,0}};  
    int sum[][]=new int [m1.length][m1[0].length];  
    System.out.println("Matrix 1 is :");  
    for (int i = 0; i <m1.length ; i++) {  
    for (int j = 0; j <m1[0].length ; j++) {  
    System.out.print(m1[i][j]+" ");  
    }  
    System.out.println();  
    }  
     
    System.out.println("Matrix 2 is :");  
    for (int i = 0; i <m2.length ; i++) {  
    for (int j = 0; j <m2[0].length ; j++) {  
    System.out.print(m2[i][j]+" ");  
    }  
    System.out.println();  
    }  
    for (int i = 0; i <sum.length ; i++) {  
    for (int j = 0; j <sum[0].length ; j++) {  
    sum[i][j]=m1[i][j]+m2[i][j];  
    }  
    }  
    System.out.println("Sum of Matrix 1 and Matrix 2 is :");  
    for (int i = 0; i <sum.length ; i++) {  
    for (int j = 0; j <sum[0].length ; j++) {  
    System.out.print(sum[i][j]+" ");  
    }  
    System.out.println();  
    }  
    }  
   }

2)

import java.util.Scanner;  
public class MatrixMultiplication {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 System.out.println("Enter length of row of matrix 1:");  
 int r1 = sc.nextInt();  
 System.out.println("Enter length of column of matrix 1:");  
 int c1 = sc.nextInt();  
 int m1[][] = new int[r1][c1];  
  
 System.out.println("Enter length of row of matrix 2:");  
 int r2 = sc.nextInt();  
 System.out.println("Enter length of column of matrix 2:");  
 int c2 = sc.nextInt();  
 int m2[][] = new int[r2][c2];  
 int mul[][]=new int[r1][c2];  
  
 if (c1 != r2) {  
 System.out.println("Matrix multiplication is not possible!");  
  
 } else {  
 System.out.println("Enter Element in matrix 1:");  
 for (int i = 0; i <r1 ; i++) {  
 for (int j = 0; j <c1 ; j++) {  
 System.out.printf("m1[%d][%d]=",i,j);  
 m1[i][j]=sc.nextInt();  
 }  
 }  
 System.out.println("\nEnter Element in matrix 2:");  
 for (int i = 0; i <r2 ; i++) {  
 for (int j = 0; j <c2 ; j++) {  
 System.out.printf("m2[%d][%d]=",i,j);  
 m2[i][j]=sc.nextInt();  
 }  
 }  
 System.out.println("\nMatrix 1 is :");  
 for (int i = 0; i < m1.length; i++) {  
 for (int j = 0; j < m1[0].length; j++) {  
 System.out.print(m1[i][j] + "\t");  
 }  
 System.out.println();  
 }  
 System.out.println("\nMatrix 2 is :");  
 for (int i = 0; i < m2.length; i++) {  
 for (int j = 0; j < m2[0].length; j++) {  
 System.out.print(m2[i][j] + "\t");  
 }  
 System.out.println();  
 }  
 for (int i = 0; i <r1 ; i++) {  
 for (int j = 0; j <c2 ; j++) {  
 mul[i][j]=0;  
// for (int k = 0; k <c1 ; k++) {  
 for (int k = 0; k <r2 ; k++) {  
 mul[i][j]+=m1[i][k]\*m2[k][j];  
 }  
 }  
 }  
 System.out.println("\nMatrix Multiplication is:");  
 for (int i = 0; i < r1; i++) {  
 for (int j = 0; j < c2; j++) {  
 System.out.print(mul[i][j] + "\t");  
 }  
 System.out.println();  
 }  
 }  
 }  
}

## 5)

class natural extends Thread  
{  
 @Override  
 public void run()  
 {  
 for (int i = 1; i <=100 ; i++) {  
 System.out.println("natural = "+i);  
 }  
 }  
}  
class even extends Thread  
{  
 @Override  
 public void run()  
 {  
 for (int i = 2; i <= 100; i+=2) {  
 System.out.println("even = "+i);  
 }  
 }  
}  
  
class prime extends Thread  
{  
 @Override  
 public void run()  
 {  
 int flag=0;  
 for (int i = 2; i <100; i+=1) {  
 for (int j = 2; j <=100 ; j++) {  
 if (i % j == 0 && i != j) {  
 flag = 1;  
 break;  
 }  
 }  
 if (flag==0)  
 {  
 System.out.println("prime = "+i);  
 }  
 flag=0;  
 }  
 }  
 }  
  
public class even\_prime\_natural {  
 public static void main(String[] args) {  
 natural n=new natural();  
 n.start();  
 prime p=new prime();  
 p.start();  
 even e=new even();  
 e.start();  
 }  
}

7)

import java.util.Scanner;  
public class linear\_linkedlist {  
 Node head=null;  
 class Node{  
 Node next;  
 int data;  
 Node(int data)  
 {  
 this.data=data;  
 this.next=null;  
 }  
 }  
 public void insert\_bottom(Scanner sc){  
 System.out.println("Enter data:");  
 int data=sc.nextInt();  
 Node new\_node=new Node(data);  
 if(head==null)  
 {  
 head=new\_node;  
 }  
 else {  
 Node temp=head;  
 while(temp.next!=null)  
 {  
 temp=temp.next;  
 }  
 temp.next=new\_node;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void insert\_top(Scanner sc){  
 System.out.println("Enter data:");  
 int data=sc.nextInt();  
 Node new\_node=new Node(data);  
 if(head==null)  
 {  
 head=new\_node;  
 }  
 else {  
 new\_node.next=head;  
 head=new\_node;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void delete\_top()  
 {  
 if(head==null)  
 {  
 System.out.println("Linkedlist is empty!!\n");  
 }  
 else {  
 head=head.next;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void delete\_bottom()  
 {  
 if(head==null)  
 {  
 System.out.println("Linkedlist is empty!!\n");  
 }  
 else {  
 System.out.println("Element Deleted!!\n");  
 if(head.next==null)  
 {  
 head=null;  
 return;  
 }  
 Node temp=head;  
 Node ptr=temp.next;  
 while (ptr.next!=null)  
 {  
 temp=ptr;  
 ptr=ptr.next;  
// temp=temp.next;  
 }  
 temp.next=null;  
 }  
 }  
 public void display()  
 {  
 System.out.println("LINEAR LINKED LIST ELEMENTS:");  
 Node temp=head;  
 while(temp!=null)  
 {  
 System.out.println(temp.data);  
 temp=temp.next;  
 }  
 System.out.println();  
 }  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 linear\_linkedlist s=new linear\_linkedlist();  
 boolean run=true;  
 System.out.println("WELCOME TO THE LINEAR LINKED LIST OPERATIONS");  
 while (run) {  
 System.out.println("1 for insert from the top\n2 for insert from the bottom\n3 for delete from the top \n4 for delete from the bottom\n5 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.insert\_top(sc);  
 case 2 ->s.insert\_bottom(sc);  
 case 3 -> s.delete\_top();  
 case 4 -> s.delete\_bottom();  
 case 5 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}