**Write the class Date having attributes like day, month & year. Add default & parameterized constructors. Add getters & setters. Add method to print the date. Add method to swap two dates**.

**class DateImpl**

**{**

**int day,month,year;**

**DateImpl()**

**{**

**//default**

**}**

**DateImpl(int day,int month,int year)**

**{**

**this.day = day;**

**this.month = month;**

**this.year = year;**

**}**

**void printDate(int day,int month,int year)**

**{**

**System.out.println(day + " " + month + " " + year);**

**}**

**public int getDay() {**

**return day;**

**}**

**public void setDay(int day) {**

**this.day = day;**

**}**

**public int getMonth() {**

**return month;**

**}**

**public void setMonth(int month) {**

**this.month = month;**

**}**

**public int getYear() {**

**return year;**

**}**

**public void setYear(int year) {**

**this.year = year;**

**}**

**}**

**public class Date {**

**public static void main(String[] args) {**

**DateImpl a = new DateImpl();**

**a.printDate(04, 06, 2021);**

**}**

**}**

* **Write a class ComplexNumber having attributes real & imaginary. Add functions like add, subtract, multiply & swap.**

**public class ComplexNumbers**

**{**

**int real = 5,img = 3;**

**void add(int real,int img)**

**{**

**System.out.println(this.real+real + " " + this.img+img);**

**}**

**void sub(int real,int img)**

**{**

**System.out.println((this.real-real) + " " + (this.img-img));**

**}**

**void mul(int real,int img)**

**{**

**System.out.println(this.real\*real + " " + this.img\*img);**

**}**

**void div(int real,int img)**

**{**

**System.out.println(this.real/real + " " + this.img/img);**

**}**

**public static void main(String[] args)**

**{**

**ComplexNumbers c = new ComplexNumbers();**

**c.add(6, 8);**

**c.sub(4, 7);**

**c.div(3, 7);**

**c.mul(9, 17);**

**}**

**}**

* **Write a class Account & add methods like deposit, withdraw, print etc.**

**public class Bank {**

**long bal = 100000;**

**long minBalance = 5000;**

**long maxTransAmount = 20000;**

**void depsoitAmount(int amount)**

**{**

**if(! (amount<maxTransAmount))**

**System.out.println("reduce your amount for deposit ");**

**bal += amount;**

**System.out.println("New Balance " + bal);**

**}**

**void withdrawAmount(int amount)**

**{**

**if((amount>maxTransAmount))**

**System.out.println("reduce your amount for withdraw");**

**long tempBal = bal - amount;**

**if((tempBal<minBalance))**

**System.out.println("cannot withdraw");**

**bal = tempBal;**

**System.out.println("New Balance " + bal);**

**}**

**public static void main(String[] args) {**

**Bank b = new Bank();**

**b.depsoitAmount(4000);**

**b.withdrawAmount(3000);**

**}**

**Write a program to implement a Stack using arrays as follows-**

* **package day4Assignments;**
* **public class Stack**
* **{**
* **int n = 100;**
* **int arr[] = new int[n];**
* **int top = -1;**
* **void push(int value)**
* **{**
* **if(top == (n-1))**
* **System.out.println("stack is full");**
* **arr[top+1] = value;**
* **top++;**
* **}**
* **void pop()**
* **{**
* **if(top < 0)**
* **System.out.println("stack is empty");**
* **top = top-1;**
* **}**
* **void display()**
* **{**
* **for(int i = 0; i<=top ; i++)**
* **{**
* **System.out.println(arr[i]);**
* **}**
* **}**
* **public static void main(String[] args) {**
* **Stack s = new Stack();**
* **s.push(61);**
* **s.push(195);**
* **s.push(51);**
* **s.push(15);**
* **s.push(41);**
* **s.push(100);**
* **s.display();**
* **System.out.println("After pop");**
* **s.pop();**
* **s.display();**


* **}**

* }
* **Write a program to implement a Queue using arrays as follows-**
* **public class Queue**
* **{**
* **int n = 100;**
* **int arr[] = new int[n];**
* **int rear=-1, front = 0;**
* **void enQueue(int value)**
* **{**
* **if(rear == (n-1))**
* **System.out.println("queue is full");**
* **arr[rear+1] = value;**
* **rear++;**
* **}**
* **void deQueue()**
* **{**
* **if(rear == front)**
* **System.out.println("queue is empty");**
* **for(int i=0;i<=rear;i++)**
* **arr[i] = arr[i+1];**
* **--rear;**
* **}**
* **void display()**
* **{**
* **for(int i = 0; i<=rear ; i++)**
* **{**
* **System.out.println(arr[i]);**
* **}**
* **}**
* **public static void main(String[] args) {**
* **Queue s = new Queue();**
* **s.enQueue(61);**
* **s.enQueue(195);**
* **s.enQueue(51);**
* **s.enQueue(15);**
* **s.enQueue(41);**
* **s.enQueue(100);**
* **s.display();**
* **System.out.println("After deQueue");**
* **s.deQueue();**
* **s.display();**


* **}**

* **}**
* **Write a single tone class. Confirm that single tone class cannot be inherited.**

**public class Singleton**

**{**

**public static void main(String[] args) {**

**Example obj = Example.getObject();**

**}**

**}**

**final class Example**

**{**

**private static Example single\_obj = null;**

**public String str ;**

**static Example obj = new Example();**

**private Example()**

**{**

**str = "Demo";**

**}**

**public static Example getObject()**

**{**

**if(single\_obj == null)**

**single\_obj = new Example();**

**System.out.println("Singleton obj created");**

**return single\_obj;**

**}**

**}**

* **Write java classes to build doubly linked list. Add functionalities like add new node, insert node, delete node, count nodes & print linked list.**
* **class DLLImpl**
* **{**
* **Node head,tail = null;**
* **class Node**
* **{**
* **int data;**
* **Node prev;**
* **Node next;**
* **Node(int d)**
* **{**
* **data = d;**
* **}**
* **}**

* **void insert(int data)**
* **{**
* **Node new\_node = new Node(data);**
* **if(head==null)**
* **{**
* **head = tail = new\_node;**
* **head.prev = null;**
* **tail.next = null;**
* **}**
* **tail.next = new\_node;**
* **new\_node.prev = tail;**
* **tail = new\_node;**
* **new\_node.next = null;**


* **}**
* **void delete(Node del)**
* **{**
* **if(head == null )**
* **{**
* **return;**
* **}**

* **if(head == del) {**
* **head = del.next;**
* **}**


* **if(del.next != null) {**
* **del.next.prev = del.prev;**
* **}**


* **if(del.prev != null) {**
* **del.prev.next = del.next;**
* **}**

* **return;**
* **}**

* **void printNodes()**
* **{**
* **Node curr = head;**
* **if(head == null)**
* **{**
* **System.out.println("DLL is empty");**
* **return;**
* **}**
* **while(curr!=null)**
* **{**
* **System.out.print(curr.data + "->" );**
* **curr = curr.next;**
* **}**
* **System.out.println(" ");**
* **}**





* **}**
* **public class DoublyLinkedList**
* **{**
* **public static void main(String[] args) {**
* **DLLImpl dl = new DLLImpl();**
* **dl.insert(10);**
* **dl.insert(20);**
* **dl.insert(30);**
* **dl.insert(40);**
* **dl.insert(50);**

* **dl.printNodes();**
* **dl.delete(dl.head.next);**
* **dl.delete(dl.tail.prev);**
* **System.out.println("after deletion");**
* **dl.printNodes();**


* **}**
* **}**