1.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.

**package** com.zensar.training;

**public** **class** Day11 {

Day11()

{

**this**.day="sunday";

**this**.month="january";

**this**.year="2021";

}

Day11(String day,String month,String year)

{

**this**.day=day;

**this**.month=month;

**this**.year=year;

}

**private** String day;

**private** String month;

**private** String year;

**public** **void** setDay(String day)

{

**this**.day=day;

}

**public** **void** setmonth(String month)

{

**this**.month=month;

}

**public** **void** setyear(String year)

{

**this**.year=year;

}

**public** String getDay()

{

**return** day;

}

**public** String getMonth()

{

**return** month;

}

**public** String getyear()

{

**return** year;

}

**public** **static** **void** printDate()

{

   Day11 day11 = **new** Day11();

    String day=day11.getDay();

    String month=day11.getMonth();

    String year=day11.getyear();

System.***out***.println("stored date : " + day+ "/" + month+"/" + year);

}

**public** **static** **void** swapDate()

{

 Day11 day11 = **new** Day11();

 String day=day11.getDay();

 String month=day11.getMonth();

    String year=day11.getyear();

 day11.setDay("thursday");

 String day2=day11.getDay();

 System.***out***.println(" before swapping = "+ day + " " + day2);

 System.***out***.println("stored date : " + day+ "/" + month+"/" + year);

 String temp;

 temp=day;

 day=day2;

 day2=temp;

 System.***out***.println(" after swapping ="+ day + " " + day2);

 System.***out***.println("stored date : " + day+ "/" + month+"/" + year);

}

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

*printDate*();

*swapDate*();

}}

Output=

stored date : sunday/january/2021

 before swapping = sunday thursday

stored date : sunday/january/2021

 after swapping =thursday sunday

stored date : thursday/january/2021

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

**package** com.zensar.training;

**public** **class** ComplexNumber {

**double** real,img;

 ComplexNumber()

 {

 }

 ComplexNumber(**double** real,**double** img){

**this**.real=real;

**this**.img=img;

 }

**public** String getValue()

 {

**return** real+"+"+img+"i";

 }

**public** **static** ComplexNumber add( ComplexNumber  c1,ComplexNumber c2)

 {

 ComplexNumber complex=**new** ComplexNumber();

 complex.real=c1.real+c2.real;

 complex.img=c1.img+c2.img;

**return** complex;

 }

**public** **static** ComplexNumber sub( ComplexNumber  c1,ComplexNumber c2)

 {

 ComplexNumber complex=**new** ComplexNumber();

 complex.real=c1.real-c2.real;

 complex.img=c1.img-c2.img;

**return** complex;

 }

**public** **static** ComplexNumber mul( ComplexNumber  c1,ComplexNumber c2)

 {

 ComplexNumber complex=**new** ComplexNumber();

 complex.real=c1.real\*c2.real;

 complex.img=c1.img\*c2.img;

**return** complex;

 }

**public** **static** ComplexNumber swap( ComplexNumber c1, ComplexNumber c2) {

 ComplexNumber complex = **new** ComplexNumber();

 System.***out***.println("Before swaping-1 =" + c1.real + "+" + c1.img + "i");

 complex.real=c1.real;

 c1.real=c1.img;

 c1.img=complex.real;

 System.***out***.println("after swaping-1= " + c1.real + "+" + c1.img + "i");

 complex.real=c2.real;

 c2.real=c2.img;

 c2.img=complex.real;

 System.***out***.println("Before swaping-2 = " + c2.real + "+" + c2.img + "i");

 System.***out***.println("after swaping-2= " + c2.real + "+" + c2.img + "i");

**return** complex;

 }

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

{

 ComplexNumber a=**new** ComplexNumber(5,6);

 ComplexNumber b=**new** ComplexNumber(7,9);

 ComplexNumber c=*add*(a,b);

 System.***out***.println("adding");

    System.***out***.println(c.getValue());

        ComplexNumber c1=*sub*(a,b);

        System.***out***.println("substraction");

    System.***out***.println(c1.getValue());

    ComplexNumber c2=*mul*(a,b);

    System.***out***.println("multiplication");

System.***out***.println(c2.getValue());

System.***out***.println("\*\*\*\*\*swap\*\*\*");

*swap*(a,b);

}

}

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

**package** com.zensar.training;

**public** **class** Account {

**private** **int** accountNumber;

**private** **double** currentBal;

      Account(){

      }

      Account(**int** accountNumber)

      {

**this**.accountNumber=accountNumber;

      }

**public** **void** setAccountNumber(**int** accountNumber)

      {

**this**.accountNumber=accountNumber;

      }

**public** **int** getAccountNumber()

      {

**return** accountNumber;

      }

**public** **void** setCurrentBal(**double** currentBal)

      {

**this**.currentBal=currentBal;

      }

**public** **double** getCurrentBal()

      {

**return** currentBal;

      }

**public**  **void** deposit(**double** amount)

      {

**this**.currentBal += amount;

      System.***out***.println("Your account is credited with rs-" +amount+"your balance is rs-" + currentBal );

      }

**public**  **void** Withdraw(**double** amount)

      {

**if**(currentBal>=amount)

      {

**this**.currentBal-=amount;

      System.***out***.println("your balance amount after dedecting rs-" + amount + "is rs-" +currentBal);

      }

**else**

      System.***out***.println("insufficient fund");

      }

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

Account account = **new** Account();

account.setAccountNumber(66167);

System.***out***.println("Your account number is :" + account.getAccountNumber());

account.setCurrentBal(10000.0);

account.deposit(5000.0);

account.Withdraw(13000.0);

}

}

4.Write a program to implement a Stack using arrays as follows-

class StackedArray {

int ary[];

push(--) { }

pop() {--) {}

}

5.Write a program to implement a Queue using arrays as follows-

class QueuedArray {

int ary[];

push(--) { }

pop() {--) {}

}

6.Write a single tone class. Confirm that single tone class cannot be inherited.

**package** com.zensar.training;

**public** **class** Singletone {

**private** **static** Singletone *singletone*=**null**;

**public** **static** Singletone singletoneMethod()

{

**if**(*singletone* == **null**)

{

System.***out***.println("object is null---create object");

*singletone*=**new** Singletone();

**return** *singletone*;

}

**else**

{

System.***out***.println("object cannot be created");

**return** *singletone*;

}

}

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

// **TODO** Auto-generated method stub

Singletone singletone=Singletone.*singletoneMethod*();

Singletone singletone1=Singletone.*singletoneMethod*();

}

}

7.Write java classes to build doubly linked list. Add functionalities like add new node, insert node, delete node, count nodes & print linked list.

class Node {

Node previous;

Node next;

Int data;

}