OOPS

LAB-11

Question1

Create the generic interface GenericQueuable<T> that contains the following abstract methods:—an abstract method insertEnd(T e) which adds the element to the Queue at end.—an abstract method removeBegin() which removes a element from the beginningof the queue.—an abstract method printQueue() which prints the queue elements from the front of the queue to end.—an abstract method isQueueEmpty() which returns true if the queue is empty otherwise return false.Here 'T' should be bounded by Person and its Children.Person–A Person contains a first name, last name, street address, zip code,and phone number. The class also includes a method that sets each datafield, and a display method that displays all of aPerson's information on a single line at the command line on the screen.CollegeEmployee–CollegeEmployee descends from Person. ACollegeEmployee also includes a Social Security number, an annual salary,and a department name, as well as methods that override the Person methodsto accept and display all CollegeEmployee data.Faculty–Faculty descends from CollegeEmployee. This class also includesa Boolean field that indicates whether the Faculty member is tenured, as wellas methods that override the CollegeEmployee methods to accept and displaythis additional piece of information.Student–Student descends from Person. In addition to the fields available inPerson, a Student contains a major field of study and a grade point average aswell as methods that override the Person methods to accept and display theseadditional facts.Create GenericQueue<T> such that it implements GenericQueuable<T>. Write a GenericQueueDemo class to test the operations of GenericQueue class with two different queues

Code:

```
import java.util.*;
interface GenericQueuable<T> { public
abstract void insertEnd(T e); public
abstract void removeBegin(); public
abstract void printQueue(); public
abstract boolean isQueueEmpty();
}
class Person
{
String firstName;
String lastName;
String address;
int zip_code;
long ph_no; void
setData()
{
Scanner sc = new Scanner(System.in);
System.out.println("Please provide the following info
!");
System.out.print("First Name - ");
 firstName = sc.next();
System.out.print("Last Name - ");
lastName = sc.next();
System.out.print("Street Address - ");
address = sc.next();
System.out.print("Zip Code - ");
```

```
zip_code = sc.nextInt();
System.out.print("Phone Number - ");
ph_no = sc.nextLong();
}
public void display()
{
 System.out.println("First Name is: "+firstName);
System.out.println("Last Name is: "+lastName);
System.out.println("Street Address is: "+address);
System.out.println("Street Zip Code is: "+zip_code);
System.out.println("Phone number is: "+ph_no);
}
}
class CollegeEmployee extends Person
{
long ssn;
double salary;
String dept;
void setData()
{
Scanner s = new Scanner(System.in);
super.setData();
System.out.println("Please enter the additional info
!");
System.out.print("Socail Security Number - "); ssn =
s.nextLong();
System.out.print("Annual Salary - $");
```

```
salary = s.nextDouble();
System.out.print("Department - "); dept
= s.next();
}
public void display()
{
 System.out.println("First Name is: "+firstName);
System.out.println("Last Name is: "+lastName);
System.out.println("Street Address is: "+address);
System.out.println("Street Zip Code is: "+zip_code);
System.out.println("Phone number is: "+ph_no);
System.out.println("Social Security Number is: "+ssn);
System.out.println("Annual Salary is: $"+salary);
System.out.println("Department is: "+dept);

}
}
class Faculty extends CollegeEmployee
{
boolean tenured; void
setData()
{
super.setData();
Scanner v = new Scanner(System.in);
System.out.println("State if the faculty member is
tenured or not :-");
String str = v.next(); if("Yesyes".indexOf(str) != -1)
tenured = true;
```

```
else tenured = false;
}
public void display()
{
if(tenured)
{
System.out.println("First Name is: "+firstName);
System.out.println("Last Name is: "+lastName);
System.out.println("Street Address is: "+address);
System.out.println("Street Zip Code is: "+zip_code);
System.out.println("Phone number is: "+ph_no);
System.out.println("Social Security Number is: "+ssn);
System.out.println("Annual Salary is: $"+salary);
System.out.println("Department is: "+dept);
System.out.println("------------Tenured--------------
");
}

else{
System.out.println("First Name is: "+firstName);
System.out.println("Last Name is: "+lastName);
System.out.println("Street Address is: "+address);
 System.out.println("Street Zip Code is: "+zip_code);
System.out.println("Phone number is: "+ph_no);
System.out.println("Social Security Number is: "+ssn);
System.out.println("Annual Salary is: $"+salary);
System.out.println("Department is: "+dept);
```

```
System.out.println("------------Not tenured-----------
");
}


}
}
class Student extends Person
{
String major;
double gpa; void
setData()
{
super.setData();
Scanner ob = new Scanner(System.in);
System.out.println("Please give additional info !");
System.out.print("Major Field of Study - ");
major = ob.next();
System.out.print("GPA - ");
gpa = ob.nextDouble();
}
public void display()
{
System.out.println("First Name is: "+firstName);
System.out.println("Last Name is: "+lastName);
System.out.println("Street Address is: "+address);
System.out.println("Street Zip Code is: "+zip_code);
System.out.println("Phone number is: "+ph_no);
```

```
 System.out.println("Major field of study is: "+major);
System.out.println("GPA is: "+gpa);

}
}
class GenericQueue<T> implements GenericQueuable<T>
{
int front, rear, capacity;
ArrayList<T> queue; public
GenericQueue(int c)
{
front = rear = 0; capacity = c;
queue = new ArrayList<T>(c);
}
public void insertEnd(T e)
{
if (capacity == rear) {
System.out.printf("\nQueue is full\n");
return;
}
else {
queue.add(rear,e);
rear++;
}
return;
}
public void removeBegin()
```

```
{
if(isQueueEmpty()) return;
else { for (int i = 0; i < rear -
1; i++) {
queue.add(i,queue.get(i+1));
}
if (rear < capacity)
queue.add(rear,null); rear--;
}
return;
}
public void printQueue()
{
System.out.println();
int i,c = 0;
if(isQueueEmpty())
return;

for (i = front; i < rear; i++) {
System.out.println((++c)+". "+queue.get(i));
}
return;
}

public boolean isQueueEmpty()
{
if (front == rear) {
```

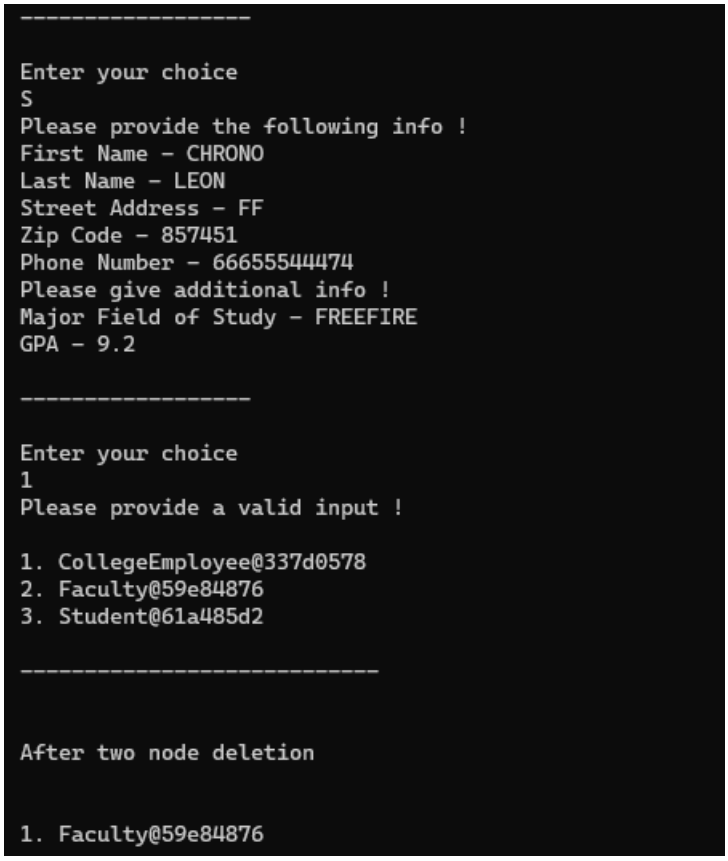
```
System.out.printf("\nQueue is Empty\n");
return true;
}
return false;
}
}
class GenericQueueDemo {
public static void main(String[] args)
{
Scanner sc = new Scanner(System.in);
for(int x=1;x<=2;x++)
{
System.out.println("\nEnter the size of Queue #"+x+" :-
");
int size = sc.nextInt();
GenericQueue<Person> q = new
GenericQueue<Person>(size);
System.out.println("You have following choices :-");
System.out.println("\n1. Press C or c for
CollegeEmployee");
System.out.println("2. Press F or f for Faculty");
System.out.println("3. Press S or s for Student");
while(size>0)
{
System.out.println("\n-----------------");
System.out.println("\nEnter your choice");
char ch = sc.next().charAt(0); switch(ch)
{
```

```
case 'C':
case 'c':
CollegeEmployee ob1 = new CollegeEmployee();
ob1.setData();
q.insertEnd(ob1);
break;
case 'F':
case 'f':
Faculty ob2 = new Faculty();
ob2.setData();
q.insertEnd(ob2);
break;
case 'S':
case 's':
Student ob3 = new Student();
ob3.setData();
q.insertEnd(ob3);
break;

default :
System.out.println("Please provide a valid input !");
}
size--;
}
q.printQueue();
q.removeBegin();
q.removeBegin();
```

```
System.out.println("\n----------------------------");
System.out.printf("\n\nAfter two node deletion\n\n");
q.printQueue();
}
}
}
```

Output:

```
C:\Users\MAJJIGA JASWANTH\Desktop\java>javac GenericQueueDemo.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>java GenericQueueDemo

Enter the size of Queue #1 :-
4
You have following choices :-

1. Press C or c for CollegeEmployee
2. Press F or f for Faculty
3. Press S or s for Student

------------------

Enter your choice
c
Please provide the following info !
First Name - Mj
Last Name - y
Street Address - vit
Zip Code - 517454
Phone Number - 9959860037
Please enter the additional info !
Socail Security Number - 8528527415
Annual Salary - $875474
Department - cse

------------------

Enter your choice
f
Please provide the following info !
First Name - M
Last Name - JASWANTH
Street Address - PUNGANUR
Zip Code - 517247
Phone Number - 7842251005
Please enter the additional info !
Socail Security Number - 6547854121
Annual Salary - $50000
Department - CSE
State if the faculty member is tenured or not :-
AP
```



Question2

Write a multithreaded program to compute sum of elements of NxN matrix. It should be done in two phases.Phase I:Create N threads to compute 'N' Columns sum, where 'i' th thread computes a 'i' th column sum.PhaseII:Create a thread to compute sum of 'N' Column's Sums. Finally,main thread is going to print result.Note:Main thread should wait till all threads completes their work.You can use Math.random() to initialize NxN matrix, in the range 0 to 100; 5<=N<=8.Implement this application in two versions, using both Thread class & Runnable interface.

Code:

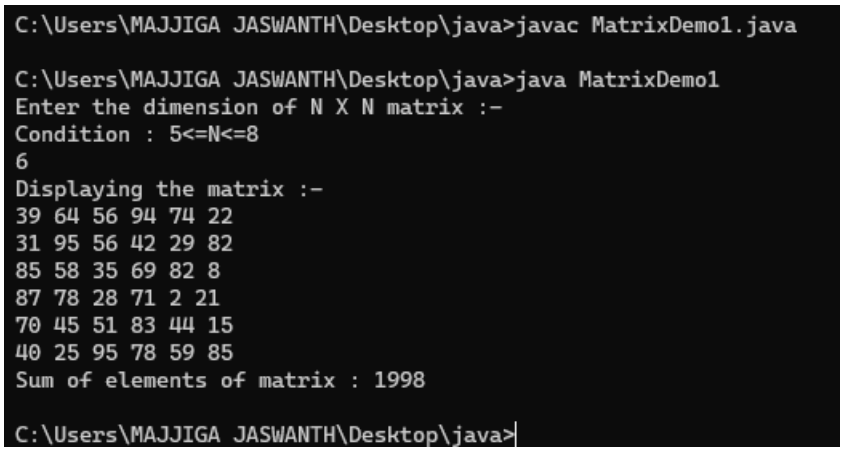
```
import java.util.*; class

Matrix1 extends Thread

{

static int[] csum;
```

```
public void run()
{
MatrixDemo1 ob = new MatrixDemo1(); csum
= new int[ob.n];
try {
for(int i=0;i<ob.n;i++)
{
for(int j=0;j<ob.n;j++)
csum[j] += ob.mat[j][i];
}

} catch(Exception e) {
System.out.println("Exception : "+e);
}
}
}
class Matrix2 extends Thread
{
MatrixDemo1 ab1 = new MatrixDemo1(); Matrix1
ab2 = new Matrix1();
static int sum = 0; public
void run()
{ try
{
for(int k=0;k<ab1.n;k++) sum
+= ab2.csum[k];
} catch(Exception e) {
```

```
System.out.println("Exception : "+e);
}
}
}
class MatrixDemo1
{
static int mat[][];
static int n; static
int z = -1;
public static void main(String[] args) throws
InterruptedException
{
Scanner sc = new Scanner(System.in);

System.out.println("Enter the dimension of N X N matrix
:-");
System.out.println("Condition : 5<=N<=8"); n =
sc.nextInt();

mat = new int[n][n];

for(int i=0;i<n;i++) for(int j=0;j<n;j++)
mat[i][j] = (int)(Math.random()*100);

System.out.println("Displaying the matrix :-");
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
```

```
System.out.print(mat[i][j]+" ");
System.out.println();
}
Matrix1 ob= new Matrix1();
ob.start();
Matrix2 obj = new Matrix2();
obj.start();
ob.join(); obj.join();
System.out.println("Sum of elements of matrix :
"+obj.sum);
}
}
```

Output:



Code:

Using runnable:

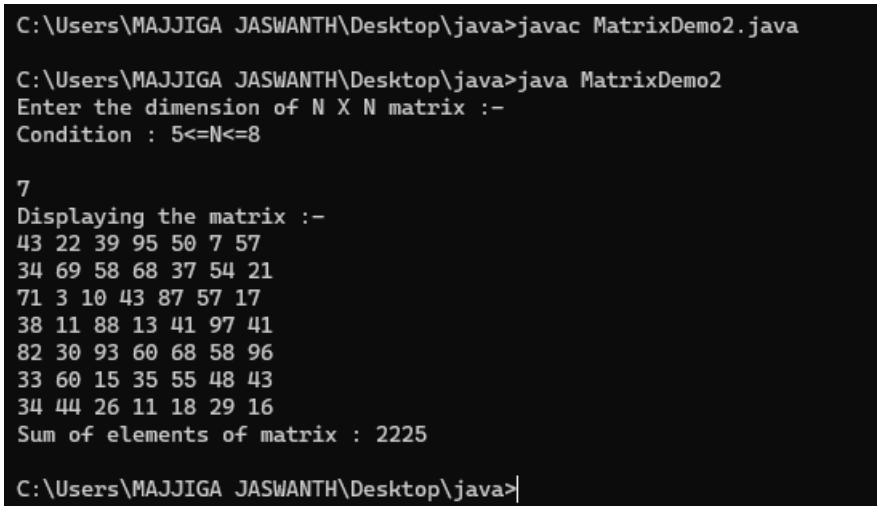
```
import java.util.*; class Matrix3
```

```
implements Runnable
{
static int[] csum;
public void run()
{
MatrixDemo2 ob = new MatrixDemo2(); csum
= new int[ob.n];
try {
for(int i=0;i<ob.n;i++)
{
for(int j=0;j<ob.n;j++)
csum[j] += ob.mat[j][i];
}

} catch(Exception e) {
System.out.println("Exception : "+e);
}
}
}
class Matrix4 implements Runnable
{
MatrixDemo2 ab1 = new MatrixDemo2();
Matrix3 ab2 = new Matrix3(); static int
sum = 0; public void run()
{ try
{
for(int k=0;k<ab1.n;k++) sum
```

```
+= ab2.csum[k];
} catch(Exception e) {
System.out.println("Exception : "+e);
}
}
}
class MatrixDemo2
{
static int mat[][]; static int n; static int z = -1;
public static void
main(String[] args) throws InterruptedException
{
Scanner sc = new Scanner(System.in);

System.out.println("Enter the dimension of N X N matrix
:-");
System.out.println("Condition : 5<=N<=8");
n = sc.nextInt();

mat = new int[n][n];

for(int i=0;i<n;i++) for(int j=0;j<n;j++)
mat[i][j] = (int)(Math.random()*100);

System.out.println("Displaying the matrix :-");
for(int i=0;i<n;i++)
{
for(int j=0;j<n;j++)
```

```
System.out.print(mat[i][j]+" ");
System.out.println();
}
Runnable obj1 = new Matrix3(); Thread
t1 = new Thread(obj1);
t1.start();
Runnable obj2 = new Matrix4(); Thread
t2 = new Thread(obj2);
t2.start();
t1.join(); t2.join();
Matrix4 ab = new Matrix4();
System.out.println("Sum of elements of matrix :
"+ab.sum);
}
}
```

Output:



Question3:

a.Create a CeaserCipher class to perform substitution and reverse substitution of characters of a message.- mEncryptionmethod –substitute a character with another

charcter of alphabet.-mDecryptionmethod -similar to mEncryptionmethod but it performsin reverse.

Each character of message is considered as numeric value with the following mapping:a-z to 0-25, respectively.(a-0,b-1,c-2,....z-25)The mEncryptionmethod replaces each character of the message with another character by using the following formula:($N(ch)+k$)%26, where $N(ch)$means Numeric value of a character 'ch', $k$means key value 0<=k<=25. The mDecryptionmethod substitutes each character with the following formula: ($N(ch)-k$)%26.Inputs to each method is a message and a key and output is substituted message printed on console character by character.(Ex: Input to mEncryptionis: rama and 25 and output is: qzlz ;Input to mDecryptionis: qzlz and 25 and output is: rama)Create a TestCeaserCipher class to test mEncryption & mDecryption methods

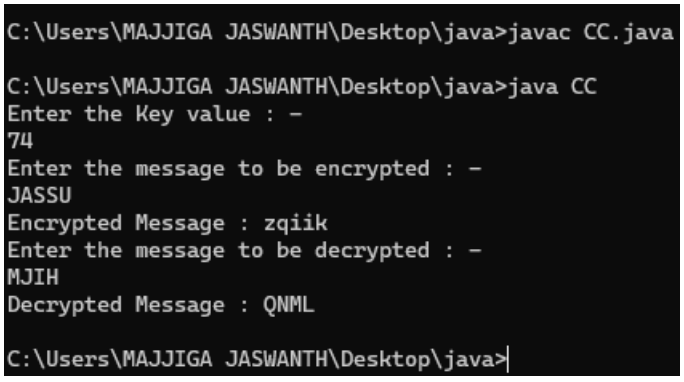
Code:

```
import java.util.*; class

CeaserCipher

{

public String mEncryption(String st, int k)

{

String str = "";

for(int i=0;i<st.length();i++)

{

char ch = st.charAt(i);

int n = (int)ch-97; n =

(n+k)%26; n += 97;

ch = (char)n; str +=

ch;
```

```
}
return str;
}
public String mDecryption(String st, int k)
{
String str = ""; for(int
i=0;i<st.length();i++)
{
char ch = st.charAt(i);
int n = (int)ch-97; n =
(n-k+26)%26; n +=
97; ch = (char)n; str
+= ch;
}
return str;
}
}
class CC
{
public static void main(String[] args)
{
CeaserCipher ob = new CeaserCipher(); Scanner sc = new
Scanner(System.in);

System.out.println("Enter the Key value : -");
int k = sc.nextInt();
```

```
System.out.println("Enter the message to be encrypted :
-");
String s = sc.next();
System.out.println("Encrypted Message :
"+ob.mEncryption(s,k));

System.out.println("Enter the message to be decrypted :
-");
s = sc.next();
System.out.println("Decrypted Message :
"+ob.mDecryption(s,k));
}
}
```

Output:



B)Jennifer comes with a message "gdhrzfnncanx". She wants to perform reverse substitution using

mDecryption method but not aware of key 'k'. To help her, develop a multithreaded program to create separate thread for each possible key 'k' and print all reverse substitutions. Do necessary changes to CeaserCipher class and provide synchronization for threads if the output from threads are mixed. Name the file as BruteForceCeaserCipher.java

Code:

```
import java.util.*; class
CeaserCipher
{
int z = 0; public String
mEncryption(String st, int k)
{
String str = ""; for(int
i=0;i<st.length();i++)
{
char ch = st.charAt(i);
int n = (int)ch-97; n =
(n+k)%26; n += 97;
ch = (char)n; str +=
ch;
}
return str;
}
synchronized String mDecryption(String st, int k)
{
String str = ""; for(int
i=0;i<st.length();i++)
```

```
{
char ch = st.charAt(i);
int n = (int)ch-97; n =
(n-k+26)%26; n +=
97; ch = (char)n; str
+= ch;
}
return str;
}
}
class MyThread1 extends Thread
{
CeaserCipher t;
int i;
String s;
MyThread1(CeaserCipher t,int i,String s)
{ this.t =
t; this.i =
i; this.s =
s;
}
public void run()
{
System.out.println((++t.z)+". When Key = "+i+",
Decrypted Message : "+t.mDecryption(s,i));
}
}
```

```
class BruteForceCeaserCipher
{
public static void main(String[] args) throws
InterruptedException
{
Scanner sc = new Scanner(System.in);
CeaserCipher ob = new CeaserCipher();

System.out.println("\nEnter the message to be decrypted
: -");
String s = sc.next();

for(int i=0;i<=25;i++)
{
MyThread1 t1=new MyThread1(ob,i,s);
t1.start();
t1.join();
}
}
}
```

Output:

```
C:\Users\MAJJIGA JASWANTH\Desktop\java>javac BruteForceCeaserCipher.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>java BruteForceCeaserCipher

Enter the message to be decrypted : -
imcoming
1. When Key = 0, Decrypted Message : imcoming
2. When Key = 1, Decrypted Message : hlbnlhmf
3. When Key = 2, Decrypted Message : gkamkgle
4. When Key = 3, Decrypted Message : fjzljfkd
5. When Key = 4, Decrypted Message : eiykiejc
6. When Key = 5, Decrypted Message : dhxjhdib
7. When Key = 6, Decrypted Message : cgwigcha
8. When Key = 7, Decrypted Message : bfvhfbgz
9. When Key = 8, Decrypted Message : aeugeafy
10. When Key = 9, Decrypted Message : zdtfdzex
11. When Key = 10, Decrypted Message : ycsecydw
12. When Key = 11, Decrypted Message : xbrdbxcv
13. When Key = 12, Decrypted Message : waqcawbu
14. When Key = 13, Decrypted Message : vzpbzvat
15. When Key = 14, Decrypted Message : uyoayuzs
16. When Key = 15, Decrypted Message : txnzxtyr
17. When Key = 16, Decrypted Message : swmywsxq
18. When Key = 17, Decrypted Message : rvlxvrwp
19. When Key = 18, Decrypted Message : qukwuqvo
20. When Key = 19, Decrypted Message : ptjvtpun
21. When Key = 20, Decrypted Message : osiusotm
22. When Key = 21, Decrypted Message : nrhtrnsl
23. When Key = 22, Decrypted Message : mqgsqmrk
24. When Key = 23, Decrypted Message : lpfrplqj
25. When Key = 24, Decrypted Message : koeqokpi
26. When Key = 25, Decrypted Message : jndpnjoh

C:\Users\MAJJIGA JASWANTH\Desktop\java>
```