**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| S.no. | TOPIC | DATE | TEACHER’S SIGN |
| 1. | Write a java program in java to add two 2d matrices. | 14/01/23 |  |
| 2. | Write a java program in java to multiply two 2d matrices. | 14/01/23 |  |
| 3. | Write a JAVA program to count numbers of vowels present in a string. | 14/01/23 |  |
| 4. | Write a JAVA program to check whether a string is palindrome or not. | 14/01/23 |  |
| 5. | Write a program to create 3 threads as-1- Natural Number as first thread :- Print 1 to 100 Natural numbers.2- Even number as second thread:- Print 1 to 100 even numbers3- Prime numbers as third thread: - Print 1 to 100 prome numbers. | 14/01/23 |  |
| 6. | Write a java program to implement basic operation(insert,delete and display) of the following using array. **a)Stack:**  **b)Queue:**  **c)CircularQueue:**  **d)deQueue:** | 14/01/23 |  |
| 7. | Write a JAVA program to implement basic operation (insert, delete and display) of Single Linked List. | 14/01/23 |  |
| 8.. | Write a JAVA program to implement basic operation (insert, delete and display) of the following using Single Linked List:- **a)Stack:**  **b)Queue:**  **c)CircularQueue:**  **d)deQueue:** . | 14/01/23 |  |

**LAB ASSIGNMENT II**

## Write a java program in java to add two 2d matrices.

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();  
 }  
 }  
}

###### 

# Output:

Matrix 1 is :

1 2 3

1 2 4

Matrix 2 is :

1 2 5

6 7 8

Sum of Matrix 1 and Matrix 2 is :

2 4 8

7 9 12

## Write a java program in java to multiply two 2d matrices.

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();  
 }  
  
 }  
 }  
}

# Output:

*Enter length of row of matrix 1:*

*2*

*Enter length of column of matrix 1:*

*3*

*Enter length of row of matrix 2:*

*3*

*Enter length of column of matrix 2:*

*2*

*Enter Element in matrix 1:*

*m1[0][0]=1*

*m1[0][1]=2*

*m1[0][2]=23*

*m1[1][0]=3*

*m1[1][1]=2*

*m1[1][2]=3*

*Enter Element in matrix 2:*

*m2[0][0]=2*

*m2[0][1]=3*

*m2[1][0]=43*

*m2[1][1]=2*

*m2[2][0]=2*

*m2[2][1]=1*

*Matrix 1 is :*

*1 2 23*

*3 2 3*

*Matrix 2 is :*

*2 3*

*43 2*

*2 1*

*Matrix Multiplication is:*

*134 30*

*98 16*

## Write a JAVA program to count numbers of vowels present in a string.

# Output:

## Write a JAVA program to check whether a string is palindrome or not.

# Output:

## Write a program to create 3 threads as-

#### 1- Natural Number as first thread :- Print 1 to 100 Natural numbers.

#### 2- Even number as second thread:- Print 1 to 100 even numbers

#### 3- Prime numbers as third thread: - Print 1 to 100 prome numbers.

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();  
 }  
}

# Output:

natural = 1

prime = 2

prime = 3

even = 2

natural = 2

natural = 3

natural = 4

natural = 5

natural = 6

natural = 7

natural = 8

prime = 5

prime = 7

prime = 11

prime = 13

prime = 17

prime = 19

prime = 23

prime = 29

prime = 31

prime = 37

prime = 41

even = 4

even = 6

even = 8

even = 10

even = 12

even = 14

even = 16

even = 18

natural = 9

prime = 43

even = 20

even = 22

even = 24

even = 26

even = 28

even = 30

even = 32

even = 34

even = 36

even = 38

even = 40

natural = 10

prime = 47

even = 42

even = 44

even = 46

even = 48

natural = 11

natural = 12

natural = 13

natural = 14

natural = 15

natural = 16

natural = 17

natural = 18

natural = 19

prime = 53

even = 50

natural = 20

natural = 21

natural = 22

natural = 23

natural = 24

natural = 25

natural = 26

natural = 27

natural = 28

natural = 29

natural = 30

natural = 31

natural = 32

prime = 59

even = 52

even = 54

even = 56

even = 58

even = 60

natural = 33

natural = 34

prime = 61

even = 62

prime = 67

natural = 35

natural = 36

natural = 37

natural = 38

natural = 39

natural = 40

natural = 41

natural = 42

natural = 43

natural = 44

even = 64

even = 66

even = 68

even = 70

even = 72

even = 74

even = 76

even = 78

even = 80

prime = 71

natural = 45

natural = 46

natural = 47

natural = 48

natural = 49

even = 82

even = 84

even = 86

even = 88

even = 90

even = 92

even = 94

even = 96

even = 98

prime = 73

prime = 79

prime = 83

prime = 89

prime = 97

natural = 50

even = 100

natural = 51

natural = 52

natural = 53

natural = 54

natural = 55

natural = 56

natural = 57

natural = 58

natural = 59

natural = 60

natural = 61

natural = 62

natural = 63

natural = 64

natural = 65

natural = 66

natural = 67

natural = 68

natural = 69

natural = 70

natural = 71

natural = 72

natural = 73

natural = 74

natural = 75

natural = 76

natural = 77

natural = 78

natural = 79

natural = 80

natural = 81

natural = 82

natural = 83

natural = 84

natural = 85

natural = 86

natural = 87

natural = 88

natural = 89

natural = 90

natural = 91

natural = 92

natural = 93

natural = 94

natural = 95

natural = 96

natural = 97

natural = 98

natural = 99

natural = 100

## Write a java program to implement basic operation(insert,delete and display) of the following using array.

**a)Stack:**

import java.util.Scanner;  
class stack{  
 int a[]=new int[4];  
 int top=-1;  
 public void pop()  
 {  
 if(top==-1)  
 {  
 System.out.println("Underflow!\n");  
 }  
 else {  
 top=top-1;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void push(Scanner sc)  
 {  
 if(top==a.length-1)  
 {  
 System.out.println("Overflow!\n");  
 }  
 else {  
 System.out.println("Enter a element:");  
 int ele= sc.nextInt();  
 top++;  
 a[top]=ele;  
 System.out.println("Element Inserted!!\n");  
 }  
 }  
 public int peek() {  
 if (top == -1) {  
 return -1;  
 } else {  
 return a[top];  
 }  
 }  
 public void display()  
 {  
 System.out.println("Stack Elements:");  
 for (int i = top; i >=0 ; i--) {  
// for (int i = top; i <a.length ; i++) {  
 System.out.println(a[i]);  
 }  
 System.out.println();  
 }  
 }  
public class stack\_using\_array {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 System.out.println("WELCOME TO THE STACK OPERATIONS");  
 stack s = new stack();  
 boolean run=true;  
 while (run) {  
 System.out.println("1 for push\n2 for pop \n3 for display\n4 for peek \nPRESS ANY KEY TO BREAKING FROM THE LOOP!! ");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.push(sc);  
 case 2 -> s.pop();  
 case 3 -> s.display();  
 case 4 ->  
 {  
 System.out.println("STACK TOP :"+s.peek()+"\n");  
 }  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE STACK OPERATIONS*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*2*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*3*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*4*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*4*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Overflow!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*4*

*STACK TOP :4*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack Elements:*

*4*

*4*

*3*

*2*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack Elements:*

*4*

*3*

*2*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Underflow!*

*1 for push*

*2 for pop*

*3 for display*

*4 for peek*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*21*

*Breaking from the loop!!*

**b) Queue:**

import java.util.Scanner;  
class linear\_queue {  
 int a[]=new int[4];  
 int f=-1,r=-1;  
 public void dequeue()  
 {  
 if(f==-1 && r==-1 )  
 {  
 System.out.println("Underflow!\n");  
 }  
 else {  
 f=f+1;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void enqueue(Scanner sc)  
 {  
 if(r==a.length-1)  
 {  
 System.out.println("Overflow!\n");  
 return;  
 }  
 System.out.println("Enter a element:");  
 int ele= sc.nextInt();  
 if(f==-1 && r==-1) {  
 f++;  
 r++;  
 a[f]=ele;  
 }  
 else {  
 r++;  
 a[r]=ele;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void display()  
 {  
 System.out.println("Queue Elements:");  
 for (int i = f; i <=r ; i++) {  
 System.out.println(a[i]);  
 }  
 System.out.println();  
 }  
}  
public class linear\_queue\_using\_array{  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 linear\_queue s=new linear\_queue();  
 boolean run=true;  
 System.out.println("WELCOME TO THE LINEAR QUEUE OPERATIONS");  
 while (run) {  
 System.out.println("1 for enqueue\n2 for dequeue\n3 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.enqueue(sc);  
 case 2 -> s.dequeue();  
 case 3 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE LINEAR QUEUE OPERATIONS*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Underflow!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*21*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*23*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Overflow!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*12*

*21*

*23*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*21*

*23*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*21*

*23*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*12*

*Breaking from the loop!!*

**c)Circular Queue:**   
import java.util.Scanner;  
class circular\_queue {  
 int a[]=new int[4];  
 int f=-1,r=-1;  
 public void dequeue()  
 {  
 if(f==-1 && r==-1 )  
 {  
 System.out.println("Underflow!\n");  
 }  
 else if (f==r)  
 {  
 System.out.println("Element Deleted!!\n");  
 f=r=-1;  
 }  
 else {  
 f=(f+1)%a.length;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void enqueue(Scanner sc)  
 {  
 if(f==(r+1)%a.length)  
 {  
 System.out.println("Overflow!\n");  
 return;  
 }  
 System.out.println("Enter a element:");  
 int ele= sc.nextInt();  
 if(f==-1 && r==-1) {  
 f++;  
 r++;  
 a[f]=ele;  
 }  
 else {  
 r=(r+1)%a.length;  
 a[r]=ele;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void display()  
 {  
 if(f==-1 && r==-1)  
 {  
 System.out.println("NO ELEMENT IN THE QUEUE!!\n");  
 return;  
 }  
 int i;  
 System.out.println("Queue Elements:");  
 for (i = f; i !=r ; i=(i+1)%a.length) {  
 System.out.println(a[i]);  
 }  
 System.out.println(a[i]+"\n");  
 }  
}  
public class ciircular\_queue\_array {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 circular\_queue s=new circular\_queue();  
 boolean run=true;  
 System.out.println("WELCOME TO THE CIRCULAR QUEUE OPERATIONS");  
 while (run) {  
 System.out.println("1 for enqueue\n2 for dequeue\n3 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.enqueue(sc);  
 case 2 -> s.dequeue();  
 case 3 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE CIRCULAR QUEUE OPERATIONS*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*21*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*21*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Overflow!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*21*

*21*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*21*

*21*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*NO ELEMENT IN THE QUEUE!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*23*

*Breaking from the loop!!*

**d)Dequeue**

import java.util.Scanner;  
class double\_ended\_queue {  
 int a[]=new int[4];  
 int f=-1,r=-1;  
 public void dequeuefront()  
 {  
 if(f==-1 && r==-1 )  
 {  
 System.out.println("Underflow!!\n");  
 }  
 else if(f==r)  
 {  
 System.out.println("Underflow!!\n");  
 f=-1;  
 r=-1;  
 }  
 else if(f==a.length-1)  
 {  
 f=0;  
 System.out.println("Element Deleted!!\n");  
 }  
 else {  
 f++;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void dequeuerear()  
 {  
 if(f==-1 && r==-1 )  
 {  
 System.out.println("Underflow!!\n");  
 }  
 else if(f==r)  
 {  
 System.out.println("Underflow!!\n");  
 f=r=-1;  
 }  
 else if(r==0) {  
 r=a.length-1;  
 System.out.println("Element Deleted!!\n");  
 }  
 else {  
 r=r-1;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void enqueuefront(Scanner sc) {  
 if (f == r + 1 || (f == 0 && r == a.length - 1)) {  
 System.out.println("Overflow!!\n");  
 } else {  
 System.out.println("Enter a element:");  
 int ele = sc.nextInt();  
 if (f == -1 && r == -1) {  
 f++;  
 r++;  
 a[f] = ele;  
 } else if (f==0) {  
 f=a.length-1;  
 a[f] = ele;  
 } else {  
 f=f-1;  
 a[f] = ele;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 }  
 public void enqueuerear(Scanner sc) {  
 if (f == r + 1 || (f == 0 && r == a.length - 1)) {  
 System.out.println("Overflow!!\n");  
 } else {  
 System.out.println("Enter a element:");  
 int ele = sc.nextInt();  
 if (f == -1 && r == -1) {  
 f++;  
 r++;  
 a[f] = ele;  
 } else if (r == a.length - 1) {  
 r = 0;  
 a[r] = ele;  
 } else {  
 r = r + 1;  
 a[r] = ele;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 }  
 public void display()  
 {  
 int i;  
 System.out.println("Queue Elements:");  
 for (i = f; i !=r ; i=(i+1)%a.length) {  
 System.out.println(a[i]);  
 }  
 System.out.println(a[i]+"\n");  
 }  
}  
public class double\_ended\_queue\_using\_array{  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 double\_ended\_queue s=new double\_ended\_queue();  
 boolean run=true;  
 System.out.println("WELCOME TO THE DEQUEUE OPERATIONS");  
 while (run) {  
 System.out.println("1 for enqueue\_front\n2 for enqueue\_rear\n3 for dequeue\_front\n4 for dequeue\_rear\n5 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.enqueuefront(sc);  
 case 2 ->s.enqueuerear(sc);  
 case 3 -> s.dequeuefront();  
 case 4 -> s.dequeuerear();  
 case 5 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE DEQUEUE OPERATIONS*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*4*

*Element Inserted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*5*

*Queue Elements:*

*4*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a element:*

*3*

*Element Inserted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*5*

*Queue Elements:*

*3*

*4*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Enter a element:*

*6*

*Element Inserted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*5*

*Queue Elements:*

*3*

*4*

*6*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Enter a element:*

*1*

*Element Inserted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Element Deleted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*5*

*Queue Elements:*

*4*

*6*

*1*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*4*

*Element Deleted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*5*

*Queue Elements:*

*4*

*6*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Element Deleted!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*4*

*Underflow!!*

*1 for enqueue\_front*

*2 for enqueue\_rear*

*3 for dequeue\_front*

*4 for dequeue\_rear*

*5 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*23*

*Breaking from the loop!!*

## Write a JAVA program to implement basic operation (insert, delete and display) of Single Linked List.

# Output:

## Write a JAVA program to implement basic operation (insert, delete and display) of the following using Single Linked List:-

**a)Stack:**

import java.util.Scanner;  
public class stack\_using\_linkedlist {  
 Node top = null;  
 class Node {  
  
 int data;  
 Node next;  
  
 public Node(int data) {  
 this.data = data;  
 this.next = null;  
 }  
 }  
  
 public void push(Scanner sc)  
 {  
  
 System.out.println("Enter a data:");  
 int data =sc.nextInt();  
 Node new\_node=new Node(data);  
 if(top==null)  
 {  
 top=new\_node;  
 }  
 else {  
 new\_node.next=top;  
 top=new\_node;  
 }  
 System.out.println("Element Inserted!!\n");  
  
 }  
 public void pop()  
 {  
 if(top==null)  
 {  
 System.out.println("Underflow!!\n");  
 }  
 else {  
 top=top.next;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void display()  
 {  
 System.out.println("Stack elements:");  
 Node temp=top;  
 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);  
 System.out.println("WELCOME TO THE STACK OPERATIONS");  
 stack\_using\_linkedlist s=new stack\_using\_linkedlist();  
 boolean run=true;  
 while (run) {  
 System.out.println("1 for push\n2 for pop \n3 for display\n PRESS ANY KEY TO BREAKING FROM THE LOOP!! ");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.push(sc);  
 case 2 -> s.pop();  
 case 3 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE STACK OPERATIONS*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Underflow!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a data:*

*12*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a data:*

*1*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a data:*

*4*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a data:*

*3*

*Element Inserted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack elements:*

*3*

*4*

*1*

*12*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack elements:*

*4*

*1*

*12*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack elements:*

*12*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Stack elements:*

*1 for push*

*2 for pop*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*121*

*Breaking from the loop!!*

**b) Queue:**

import java.util.Scanner;  
  
public class linear\_queue\_using\_linkedlist {  
 Node f=null;  
 Node r=null;  
 class Node{  
 Node next;  
 int data;  
 Node(int data)  
 {  
 this.data=data;  
 this.next=null;  
 }  
 }  
  
 public void enqueue(Scanner sc)  
 {  
 System.out.println("Enter a number:");  
 int data=sc.nextInt();  
 Node new\_node=new Node(data);  
 if(f==null)  
 {  
 f=new\_node;  
 r=new\_node;  
 }  
 else {  
 r.next=new\_node;  
 r=new\_node;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void dequeue()  
 {  
 if(f==null)  
 {  
 System.out.println("Underflow!!\n");  
 }  
 else {  
 f=f.next;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void display()  
 {  
 System.out.println("Queue Elements:");  
 Node temp=f;  
 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\_queue\_using\_linkedlist s=new linear\_queue\_using\_linkedlist();  
 boolean run=true;  
 System.out.println("WELCOME TO THE LINEAR QUEUE OPERATIONS");  
 while (run) {  
 System.out.println("1 for enqueue\n2 for dequeue\n3 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.enqueue(sc);  
 case 2 -> s.dequeue();  
 case 3 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE LINEAR QUEUE OPERATIONS*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*2*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*4*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*2*

*4*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*2*

*4*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*4*

*12*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Underflow!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*212*

*Breaking from the loop!!*

**c) Circular Queue.**

import java.util.Scanner;  
public class circular\_queue\_using\_linkedlist {  
 Node f=null;  
 Node r=null;  
 class Node{  
 Node next;  
 int data;  
 Node(int data)  
 {  
 this.data=data;  
 this.next=null;  
 }  
 }  
  
 public void enqueue(Scanner sc)  
 {  
 System.out.println("Enter a number:");  
 int data=sc.nextInt();  
 Node new\_node=new Node(data);  
 if(f==null)  
 {  
 f=new\_node;  
 r=new\_node;  
// r.next=f;  
 }  
 else {  
 r.next=new\_node;  
 r=new\_node;  
 new\_node.next=f;  
 }  
 System.out.println("Element Inserted!!\n");  
 }  
 public void dequeue()  
 {  
 if(f==null)  
 {  
 System.out.println("Underflow!!\n");  
 }  
 else if(f==r)  
 {  
 System.out.println("Element Deleted!!\n");  
 f=r=null;  
 }  
 else {  
  
 f=f.next;  
 r.next=f;  
 System.out.println("Element Deleted!!\n");  
 }  
 }  
 public void display()  
 {  
 if (f==null)  
 {  
 System.out.println("No Element in the Queue!!\n");  
 return;  
 }  
 System.out.println("Queue Elements:");  
 Node temp=f;  
 while (temp.next!=f)  
 {  
 System.out.println(temp.data);  
 temp=temp.next;  
 }  
 System.out.println(temp.data);  
 System.out.println();  
 }  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
 circular\_queue\_using\_linkedlist s=new circular\_queue\_using\_linkedlist();  
 boolean run=true;  
 System.out.println("WELCOME TO THE CIRCULAR QUEUE OPERATIONS");  
 while (run) {  
 System.out.println("1 for enqueue\n2 for dequeue\n3 for display\nPRESS ANY KEY TO BREAKING FROM THE LOOP!!");  
 int n = sc.nextInt();  
 switch (n) {  
 case 1 -> s.enqueue(sc);  
 case 2 -> s.dequeue();  
 case 3 -> s.display();  
 default -> {  
 System.out.println("Breaking from the loop!!");  
 run=false;  
 }  
 }  
 }  
 }  
}

# Output:

*WELCOME TO THE CIRCULAR QUEUE OPERATIONS*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*3*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*3*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*1*

*Enter a number:*

*12*

*Element Inserted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*12*

*3*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*Queue Elements:*

*3*

*12*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Element Deleted!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*3*

*No Element in the Queue!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*2*

*Underflow!!*

*1 for enqueue*

*2 for dequeue*

*3 for display*

*PRESS ANY KEY TO BREAKING FROM THE LOOP!!*

*121*

*Breaking from the loop!!*

**d)Dequeue**

#### 

**BABU BANARASI DAS UNIVERSITY**

**LUCKNOW**

****

**Master Of Computer Application**

**Semester-1**

**School Of Computer Application**

**Data Structure using java**

**Lab Assignment**

**Session 2022-23**

### Submitted to:

### Mr. Shamim Ahmad Ansari

### Submitted by:

### Syed Waqarul Hasan Naqvi

### Roll No. : 53

### MCA 11