**EXPERIMENT – 3.5**

**AIM:** Create a java program to implement stack concept.

**SOURCE CODE:**

class Stack {

private int arr[];

private int top;

private int capacity;

Stack(int size) {

arr = new int[size];

capacity = size;

top = -1;

}

public void push(int x) {

if (isFull()) {

System.out.println("Stack OverFlow");

System.exit(1); }

System.out.println("Inserting " + x);

arr[++top] = x; }

public int pop() {

if (isEmpty()) {

System.out.println("STACK EMPTY");

System.exit(1); }

return arr[top--];}

public int getSize() {

return top + 1; }

public Boolean isEmpty() {

return top == -1;

}

public Boolean isFull() {

return top == capacity - 1; }

public void printStack() {

System.out.println("Top element: "+arr[top]);

System.out.println("Items:-");

for (int i = top; i >= 0; i--) {

System.out.println(arr[i]); }

}

public static void main(String[] args) {

Stack stack = new Stack(5);

System.out.println("Inserting elements in the stack:-");

stack.push(1); stack.push(2); stack.push(3);

System.out.println("The size of the stack is: " + stack.getSize());

System.out.println("Input Stack: ");

stack.printStack();

System.out.println("Popping an element from the stack");

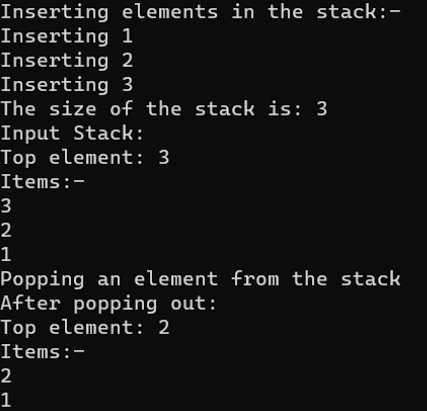
stack.pop();

System.out.println("After popping out:");

stack.printStack(); }

}

**OUTPUT:**

****

**EXPERIMENT – 3.6**

**AIM:** Create a java program to implement queue concept.

**SOURCE CODE:**

public class Queue {

int SIZE = 5;

int items[] = new int[SIZE];

int front, rear;

Queue() {

front = -1;

rear = -1; }

boolean isFull() {

if (front == 0 && rear == SIZE - 1) {

return true; }

return false; }

boolean isEmpty() {

if (front == -1)

return true;

else

return false; }

void enQueue(int element) {

if (isFull()) {

System.out.println("Queue is full"); }

else {

if (front == -1) {

front = 0; }

rear++;

items[rear] = element;

System.out.println("Inserting " + element); }

}

int deQueue() {

int element;

if (isEmpty()) {

System.out.println("Queue is empty");

return (-1); }

else {

element = items[front];

if (front >= rear) {

front = -1;

rear = -1; }

else {

front++; }

return (element); }

}

int getSize(){

return rear+1;

}

void display() {

int i;

if (isEmpty()) {

System.out.println("Empty Queue");

}

else {

System.out.println("Front index-> " + front);

System.out.print("Items -> ");

for (i = front; i <= rear; i++)

System.out.print(items[i] + " ");

System.out.println("\nRear index-> " + rear);

}

}

public static void main(String[] args) {

Queue q = new Queue();

System.out.println("Inserting elements in the Queue:");

for(int i = 1; i < 6; i ++) {

q.enQueue(i); }

System.out.println("The size of the Queue is: " + q.getSize());

System.out.println("Input Queue:");

q.display();

System.out.println("Deleting an element from the Queue");

q.deQueue();

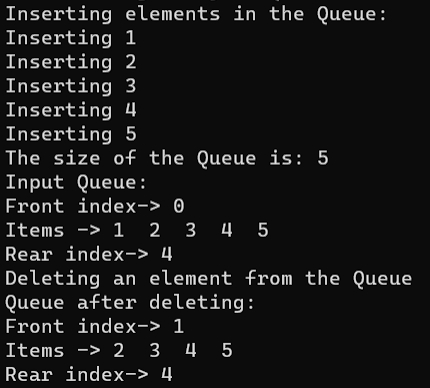
System.out.println("Queue after deleting:");

q.display();

}

}

**OUTPUT:**

****