1. Implement an ArrayDequeue and all of its methods such as add(), addFirst(), addLast(), element(), poll(), push(), remove.

package day20;

import java.util.ArrayDeque;

import java.util.Deque;

public class ArrayDequeDemo

{

public static void main(String[] args)

{

Deque<String> dq= new ArrayDeque<String>();

dq.add("to");

dq.addFirst("WELCOME");

dq.addLast("java");

System.out.println(dq);

System.out.println(dq.element());

System.out.println(dq.poll());

dq.push("amazing");

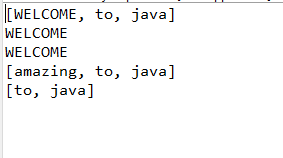
System.out.println(dq);

dq.remove();

System.out.println(dq);

}

}



1. Implement a PriorityQueue and use all the methods.

package day19;

import java.util.Iterator;

import java.util.PriorityQueue;

public class PriorityQueueDemo {

public static void main(String[] args) {

PriorityQueue<String> pq = new PriorityQueue<>();

pq.add("Ajay");

pq.add("Vijay");

pq.add("Raj");

pq.add("Gagan");

System.out.println("head:"+pq.element());

System.out.println("head:"+pq.peek());

System.out.println("iterating the queue elements:");

Iterator itr=pq.iterator();

while(itr.hasNext()){

System.out.println(itr.next());

}

pq.remove();//removes the head element

pq.poll(); //removes the head

System.out.println("after removing two elements:");

Iterator<String> itr2=pq.iterator();

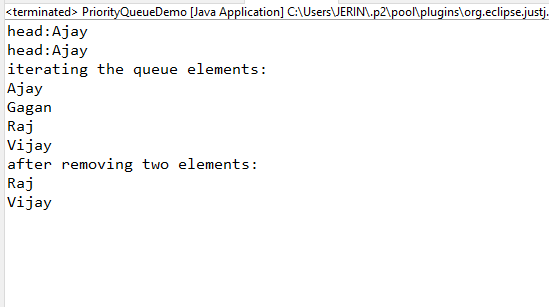
while(itr2.hasNext()){

System.out.println(itr2.next());

}

}

}



1. Implement a Stack and all of its methods peek(), push(), pop(), and to determine the size of the stack.

package day18;

import java.util.Scanner;

public class stackOp

{

int STACK\_SIZE;

int[] stack;

public stackOp(int n)

{

this.STACK\_SIZE = n;

this.stack = new int[n];

}

public int push(int top,int ele)

{

if(top == STACK\_SIZE-1)

{

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

}

else

{

top++;

stack[top] = ele;

}

return top;

}

public int pop(int top)

{

if (top == -1)

{

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

}

else

{

System.out.println(String.format("%d removed from the Stack.",stack[top--]));

}

return top;

}

public void display(int top)

{

if (top == -1)

{

System.out.println("Stack is Empty.");

}else

{

while(top>0)

{

System.out.print(String.format("%d ",stack[top]));

top--;

}

System.out.print(String.format(" %d",stack[top]));

}

}

public int stackSize(int top)

{

int count = 0;

int temp = top;

if (temp == -1)

{

return count;

}else

{

while (temp>= 0)

{

++count;

temp--;

}

}

return count;

}

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter the Size of the Stack : ");

int n = sc.nextInt();

stackOp stack=new stackOp(n);

int top = -1;

System.out.println("Enter the Stack elements : ");

for (int i = 0;i<n;i++)

{

int ele = sc.nextInt();

top = stack.push(top,ele);

}

System.out.println("Stack after Inserting the elements : ");

stack.display(top);

System.out.println();

System.out.println("Stack Size after inserting the elements : ");

System.out.print(stack.stackSize(top));

System.out.println();

System.out.println("After applying the pop() operation :");

top = stack.pop(top);

System.out.println("Stack after 1st pop() operation performed : ");

stack.display(top);

System.out.println();

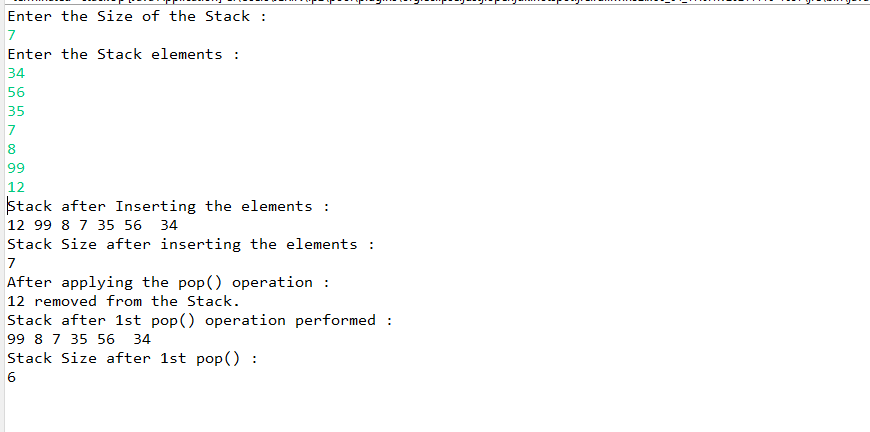
System.out.println("Stack Size after 1st pop() : ");

System.out.print(stack.stackSize(top));

System.out.println();

}

}



1. Write a program to implement insertion sort.

package day16;

public class InsertionSort

{

public static void insertionSort(int array[])

{

int n = array.length;

for (int j = 1; j < n; j++)

{

int key = array[j];

int i = j-1;

while ( (i > -1) && ( array [i] > key ) )

{

array [i+1] = array [i];

i--;

}

array[i+1] = key;

}

}

public static void main(String[] args)

{

int[] arr1 = {9,145,31,2,13,10,58,22};

System.out.println("Before Insertion Sort");

for(int i:arr1)

{

System.out.print(i+" ");

}

System.out.println();

insertionSort(arr1);//sorting array using insertion sort

System.out.println("After Insertion Sort");

for(int i:arr1)

{

System.out.print(i+" ");

}

}

}

