**6. Implement the following Data structures in Java ?**

**a)Linked Lists b) Stacks c) Queues d) Set e) Map**

**Linked List**

program:

import java.util.\*;

public class LinkedListDemo {

public static void main(String args[]) {

// create a linked list

LinkedListll = new LinkedList();

* add elements to the linked list ll.add("F");

ll.add("B");

ll.add("D");

ll.add("E");

ll.add("C");

ll.addLast("Z");

ll.addFirst("A"); ll.add(1, "A2");

System.*out*.println("Original contents of ll: " + ll);

* remove elements from the linked list ll.remove("F");

ll.remove(2);

System.*out*.println("Contents of ll after deletion: " + ll);

* remove first and last elements ll.removeFirst(); ll.removeLast();

System.*out*.println("ll after deleting first and last: "+ ll);

* get and set a value Object val = ll.get(2);

ll.set(2, (String) val + " Changed"); System.*out*.println("ll after change: " + ll);

}

}

Output:

Original contents of ll: [A, A2, F, B, D, E, C, Z]

Contents of ll after deletion: [A, A2, D, E, C, Z]

ll after deleting first and last: [A2, D, E, C]

ll after change: [A2, D, E Changed, C]

**b) Stacks**

import java.util.\*;

public class StackDemo {

static void showpush(Stack st, int a) {

st.push(new Integer(a));

System.out.println("push(" + a + ")");

System.out.println("stack: " + st);

}

static void showpop(Stack st) {

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

Integer a = (Integer) st.pop();

System.out.println(a);

System.out.println("stack: " + st);

}

public static void main(String args[]) {

Stack st = new Stack();

System.out.println("stack: " + st);

showpush(st, 42);

showpush(st, 66);

showpush(st, 99);

showpop(st);

showpop(st);

showpop(st);

try {

showpop(st);

}catch (EmptyStackException e) {

System.out.println("empty stack");

}

}

}

output:

stack: [ ]

push(42)

stack: [42]

push(66)

stack: [42, 66]

push(99)

stack: [42, 66, 99]

pop -> 99

stack: [42, 66]

pop -> 66

stack: [42]

pop -> 42

stack: [ ]

pop -> empty stack

**c) Queues**

* import java.util.LinkedList;

import java.util.Queue; public class QueueExample

{

public static void main(String[] args)

{

Queue<Integer> q = new LinkedList<>();

* + Adds elements {0, 1, 2, 3, 4} to queue

for (int i=0; i<5; i++)

q.add(i);

* Display contents of the queue. System.out.println("Elements of queue-"+q);
  + To remove the head of queue.

intremovedele = q.remove(); System.out.println("removed element-" + removedele); System.out.println(q);

* To view the head of queue int head = q.peek();
* Rest all methods of collection interface,
* Like size and contains can be used with this
* implementation.

int size = q.size();

System.out.println("Size of queue-" + size);

}

}

Output:

Elements of queue-[0, 1, 2, 3, 4]

removed element-0

[1, 2, 3, 4]

head of queue-1

Size of queue-4

**d) Set**

public class SetDemo {

public static void main(String args[]) {

int count[] = {34, 22,10,60,30,22};

Set<Integer> set = new HashSet<Integer>();

try{

for(int i = 0; i<5; i++){

set.add(count[i]);

}

System.*out*.println(set);

TreeSetsortedSet = new TreeSet<Integer>(set);

System.*out*.println("The sorted list is:");

System.*out*.println(sortedSet);

System.*out*.println("The First element of the set is: "+(Integer)sortedSet.first());

System.*out*.println("The last element of the set is: "+(Integer)sortedSet.last());

}

catch(Exception e){}

}

}

Output:

[34, 22, 10, 60, 30]

The sorted list is:

[10, 22, 30, 34, 60]

The First element of the set is: 10

The last element of the set is: 60

**e) Map**

Program:

import java.awt.Color;

import java.util.HashMap;

import java.util.Map;

import java.util.Set;

public class MapDemo

{

public static void main(String[] args)

{

Map<String, Color>favoriteColors = new HashMap<String, Color>(); favoriteColors.put("sai", Color.*BLUE*); favoriteColors.put("Ram", Color.*GREEN*); favoriteColors.put("krishna", Color.*RED*);

favoriteColors.put("narayana", Color.*BLUE*); // Print all keys and values in the map

Set<String>keySet = favoriteColors.keySet(); for (String key : keySet)

{

Color value = favoriteColors.get(key);

System.*out*.println(key + " : " + value);

}

}

}

Output:

narayana : java.awt.Color[r=0,g=0,b=255]

sai : java.awt.Color[r=0,g=0,b=255]

krishna : java.awt.Color[r=255,g=0,b=0]

Ram : java.awt.Color[r=0,g=255,b=0]