1. **Write a program to implement Binary Search?**

**Ans.**

public class BinarySearch {

public int binarySearch(int[] x, int val) {

int start = 0;

int end = x.length - 1;

while (start <= end) {

int mid = (start + end) / 2;

if (val == x[mid]) {

return mid;

}

if (val < x[mid]) {

end = mid - 1;

} else {

start = mid + 1;

}

}

return -1;

}

public static void main(String[] args) {

BinarySearch bs = new BinarySearch();

int[] arr = {10,8,99,87,58,76,3,95,110,25,75};

System.out.println("Position Of 14 : "+bs.binarySearch(arr, 14));

System.out.println("Position of 76 : "+bs.binarySearch(arr, 76));

}

}

**Output:**

Position Of 14 : -1

Position of 76 : 5

1. **Write a program to implement Bubble sort?**

**Ans.**

public class BubbleSort {

public static void bubSort(int[] x) {

int n = x.length;

int temp = 0;

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

for(int j=1; j < (n-i); j++){

if(x[j-1] > x[j]){

temp = x[j-1];

x[j-1] = x[j];

x[j] = temp;

}

}

}

}

public static void main(String[] args) {

int x[] ={10,8,99,87,58,76,3,95,110,25,75};

System.out.println("Array Before Bubble Sort");

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

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

}

System.out.println();

bubSort(x);

System.out.println("Array After Bubble Sort");

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

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

}

}

}

**Output:**

Array Before Bubble Sort

10 8 99 87 58 76 3 95 110 25 75

Array After Bubble Sort

3 8 10 25 58 75 76 87 95 99 110

1. **Write a program to implement the stack operations?**

**Ans.**

import java.util.\*;

class ArrayStack

{

protected int arr[];

protected int top, size, len;

public ArrayStack(int n)

{

size = n;

len = 0;

arr = new int[size];

top = -1;

}

public boolean isEmpty()

{

return top == -1;

}

public boolean isFull()

{

return top == size -1 ;

}

public int getSize()

{

return len ;

}

public int peek()

{

if( isEmpty() )

throw new NoSuchElementException("Underflow Exception");

return arr[top];

}

public void push(int i)

{

if(top + 1 >= size)

{

throw new IndexOutOfBoundsException("Overflow Exception");

}

else if(top + 1 < size )

{

arr[++top] = i;

len++;

}

}

public int pop()

{

if( isEmpty() )

throw new NoSuchElementException("Underflow Exception");

len-- ;

return arr[top--];

}

public void display()

{

System.out.print("\nStack = ");

if (len == 0)

{

System.out.print("Empty\n");

return ;

}

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

{

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

System.out.println();

}

}

}

class StackImplement

{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

System.out.println("Stack Test\n");

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

int n = scan.nextInt();

/\* Creating object of class ArrayStack \*/

ArrayStack as = new ArrayStack(n);

/\* Perform Stack Operations \*/

char ch;

do{

System.out.println("\nStack Operations");

System.out.println("1. push");

System.out.println("2. pop");

System.out.println("3. peek");

System.out.println("4. check empty");

System.out.println("5. check full");

System.out.println("6. size");

int choice = scan.nextInt();

switch (choice)

{

case 1 :

System.out.println("Enter integer element to push");

try

{

as.push( scan.nextInt() );

}

catch (Exception e)

{

System.out.println("Error : " + e.getMessage());

}

break;

case 2 :

try

{

System.out.println("Popped Element = " + as.pop());

}

catch (Exception e)

{

System.out.println("Error : " + e.getMessage());

}

break;

case 3 :

try

{

System.out.println("Peek Element = " + as.peek());

}

catch (Exception e)

{

System.out.println("Error : " + e.getMessage());

}

break;

case 4 :

System.out.println("Empty status = " + as.isEmpty());

break;

case 5 :

System.out.println("Full status = " + as.isFull());

break;

case 6 :

System.out.println("Size = " + as.getSize());

break;

default :

System.out.println("Wrong Entry \n ");

break;

}

as.display();

System.out.println("\nDo you want to continue (Type y or n) \n");

ch = scan.next().charAt(0);

}

while (ch == 'Y'|| ch == 'y');

}

}

**Output:**

Stack Test

Enter Size of Integer Stack

5

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

1

Enter integer element to push

12

Stack = 12

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

2

Popped Element = 2

Stack = Empty

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

6

Size = 0

Stack = Empty

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

1

Enter integer element to push

32

Stack = 32

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

1

Enter integer element to push

36

Stack = 36

32

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

6

Size = 2

Stack = 36

32

Do you want to continue (Type y or n)

y

Stack Operations

1. push

2. pop

3. peek

4. check empty

5. check full

6. size

5

Full status = false

Stack = 36

32

Do you want to continue (Type y or n)

n

1. **Write a program to search any data in given LinkList without using collections framework?**

**Ans**

**.** class Node

{

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

public class LinkedList

{

Node head;

public void push(int new\_data)

{

Node new\_node = new Node(new\_data);

new\_node.next = head;

head = new\_node;

}

public boolean search(Node head, int x)

{

if (head == null)

return false;

if (head.data == x)

return true;

return search(head.next, x);

}

public static void main(String args[])

{

LinkedList llist = new LinkedList();

llist.push(10);

llist.push(30);

llist.push(11);

llist.push(21);

llist.push(14);

if (llist.search(llist.head, 13))

System.out.println("Yes");

else

System.out.println("No");

}

void add(String f) {

throw new UnsupportedOperationException("Not supported yet.");

}

}

**Output:**

No, the number is not present

1. **Write a program to Add and Delete data inside LinkList without using collections framework?**

**Ans:**

import java.util.LinkedList;

public class LinkedListNew {

public static void main(String args[]) {

// create a linked list

LinkedList l = new LinkedList();

// add elements to the linked list

l.add("F");

l.add("B");

l.add("D");

l.add("E");

l.add("C");

l.addLast("Z");

l.addFirst("A");

l.add(1, "A2");

System.out.println("Original contents of the linked list is: " + l);

// remove elements from the linked list

l.remove("F");

l.remove(2);

System.out.println("Contents of linked list after deletion: "

+ l);

// remove first and last elements

l.removeFirst();

l.removeLast();

System.out.println("Linked List after deleting first and last: "+ l);

// get and set a value

Object val = l.get(2);

l.set(2, (String) val + " Changed");

System.out.println("Linked List after change: " + l);

}

}

**Output:**

Original contents of the linked list is: [A, A2, F, B, D, E, C, Z]

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

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

Linked List after change: [A2, D, E Changed, C]