THE COLLECTION FRAMEWORK

We use an array to store a group of elements and handle them easily. We can also store group of objects in an array

Ex:

Person obj[]=new Person[10];

In the above example the array can store 10 Person objects

But there are certain inconveniences in this mechanism . They are

Inserting and deleting elements in the middle of the array is difficult

**To process the elements there are no methods available .**

Due to these problems programmers want better mechanism to store a group of objects. That is Collection Object

Collection FrameWork:

A collection frame work is a class library to handle groups of objects.

All the collection classes are in java.util package

All the collection classes are implementation classes of different interfaces

|  |  |
| --- | --- |
| Interface Type | Implementation Class |
| Set<T> | HashSet<T> |
| LinkedHashSet<T> |
| List<T> | Stack<T> |
| LinkedList<T> |
| ArrayList<T> |
| Vector<T> |
| Queue<T> | LinkedList<T> |
| Map<T> | HashMap<T> |
| HashTable<T> |

**Sets:**

A set represents a group of elements arranged just like an aray.

A set Will not allow Duplicate elements…If we try to insert duplicate ele it will not stored into the set

**List:**

**Lists are like sets .But list allows duplicate values to be stored**

**Queues**

Queue represents arrangement of elements in FIFO(First In First Out) Order.

**Map:**

Map stores elements in the form of key and value pair ..Key should be unique value.

**RETRIEVING ELEMENTS FROM COLLECTIONS**

*There are 4 ways to retrieve the ele from a collection object*

**1 . for-each looping**

**2 . iterator interface**

**3 . ListIterator interface**

**4 . Enumeration interface**

1. **. for- each looping :**

**Which repeatedly executes a group of statements**

*For(variable: collection object)*

*{*

*Statements;*

*}*

1. **. iterator interface:**

**Iterator contains methods**

*a)Boolean hasNext()-***This method returns True if the iterator has more elements**

*b)Element next():* **This method returns the next element**

*c)void remove():***This method removes from the collection the last element returned by the iterator**

**3 . ListIterator interface**

**Methods**

1. *Boolean hasNext()*
2. *Boolean hasPrevious()*

Element Next()

1. *Element previous()*
2. *Void remove()*

**4 . Enumeration interface**

**Methos:**

**Boolean has*More*Elements():***This method tests if the enumeration has any more elements or not*

**Element nextElement():**This returns the next element

**Implementation class of Set interface**

**HashSet class**

This class represents set of objects.It does not guarantee the order of elements also it does not allow the duplicates.

**Syntax:**

Class HashSet<T>

T represents generic type perameter

**Object creation:**

**HashSet<String> hs=new HashSet<String>();**

**Methods:**

**Boolean add(obj)**..adds an element

**Boolean remove(obj)….**.removes the ele

**Void clear()**….removes all the elements

**Boolean contains(obj)**…..This method returns True if the HashSet Contains the specified Obj otherwise it returns False

**Boolean isEmpty():**This returns True if the HashSet contains no elements

**Int size()**…. This returns the number of elements present in the hash Set

import java.util.\*;

import java.io.\*;

class HS

{

public static void main(String args[])throws IOException

{

HashSet<String> hs=new HashSet<String>();

// hs.add("abc");

// hs.add("def");

// hs.add("xyz");

// hs.add("mno");

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the data into the HashSet");

for(int i=0;i<10;i++)

{

String str=br.readLine();

hs.add(str);

}

System.out.println(hs);

System.out.println("Iteams in Hash Set");

Iterator it=hs.iterator();

while(it.hasNext())

{

String str1=(String)it.next();

System.out.println(str1);

}

}

}

import java.util.\*;

class HS

{

public static void main(String args[])

{

HashSet<Integer> obj=new HashSet<Integer>();

Scanner sc=new Scanner(System.in);

int ele;

while(true)

{

System.out.println("1.Add 2.Delete 3.Contains 4.Display");

System.out.println("Enter the Choice");

int ch=sc.nextInt();

switch(ch)

{

case 1: System.out.println("Enter the elements");

ele=sc.nextInt();

obj.add(ele);

break;

case 2:System.out.println("Enter the element");

ele=sc.nextInt();

obj.remove(ele);

break;

case 3:System.out.println("Enter the element");

ele=sc.nextInt();

boolean r=obj.contains(ele);

if(r==true)

System.out.println("Found");

else

System.out.println("Not Found");

break;

case 4: Iterator it=obj.iterator();

while(it.hasNext())

{

ele=(int)it.next();

System.out.println(ele);

}

break;

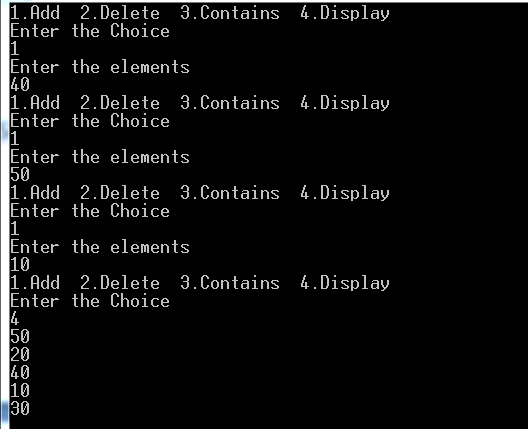
default:System.exit(0);

}

}

}

}



LIST

STACK:

A Stack represents a group of elements stored in LIFO(Last in First Out). Inserting elements into stack is called PUSH operation.Removing elements from the stack is called POP operation…

Stack class methos

Boolean empty()..This method checks whether stack is empty or not

Element peek()…This method return the top most object from the stack.

Elements pop()….pops the top elements from the stack

Element push(element object)….pushes the element in to the top of the stack

Int search(Object obj)…return the position of an element in the stack ..if not present it returns the -1

import java.io.\*;

import java.util.\*;

class StackEx

{

public static void main(String args[])throws IOException

{

Stack<Integer> st=new Stack<Integer>();

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int ele;

while(true)

{

System.out.println("1.PUSH 2.POP 3.SEARCH");

System.out.println("Enter UR choice");

int ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:System.out.println("Enter the element");

ele=Integer.parseInt(br.readLine());

st.push(ele);

System.out.println("ele succefully inserted");

break;

case 2:Integer i=st.pop();

System.out.println("Deleted ele :"+i);

break;

case 3:System.out.println("Enter the Search element");

ele=Integer.parseInt(br.readLine());

int r=st.search(ele);

if(r==-1)

System.out.println("Element not found");

else

System.out.println("Element found at"+r);

break;

default:System.exit(0);

}

}

}

}

LINKED LIST:

A Linked list contains a group of elements in the form of nodes.Each node will have three fields

1.data field: Contains Data

2.Linke Field: Contains reference to Previous Node and Next Node

Linked list is very convenient to store , Insert , delete and display data

Linked List class Methods:

Boolean add(element obj)

Void add(int pos,element obj)…

Void addFirst(element obj)

Void addLast(element obj)

Element removeFirst()

Element removeLast()

Void clear()

Element getFirst()

Element getLast()

Int size()

import java.util.\*;

import java.io.\*;

class LLDemo

{

public static void main(String args[])throws IOException

{

LinkedList<String> ls=new LinkedList<String>();

ls.add("ABC");

ls.add("DEF");

ls.add("XYZ");

ls.add("MNO");

System.out.println("LIST ele are :"+ls);

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int pos,ch=0;String ele;

while(ch<4)

{

System.out.println("List Operations are :");

System.out.println("1.Add Element 2.Remove 3.Change 4.Exit");

System.out.println("Ur Choice PLZ:");

ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:System.out.println("Enter the Element");

ele=br.readLine();

System.out.println("At Possintion");

pos=Integer.parseInt(br.readLine());

ls.add(pos-1,ele);

break;

case 2:

System.out.println("Enter Pos");

pos=Integer.parseInt(br.readLine());

String str=ls.remove(pos-1);

System.out.println("Removed ele is :"+str);

break;

case 3:

System.out.println("Enter Pos");

pos=Integer.parseInt(br.readLine());

System.out.println("Element PLZ:");

ele=br.readLine();

ls.set(pos-1,ele);

break;

default: return;

}

System.out.println(ls);

}

}

}

**ArrayList:**

An array list is like as an array which can grow in memory dynamically.

**Array List Class Methods:**

Boolean add(element obj)

Void add (int pos ,element obj)

Boolen remove(Object obj)

Void clear()

Boolean contains(Object obj)

Int size()

import java.util.\*;

//import java.io.\*;

class AList

{

public static void main(String args[])

{

ArrayList<String> ar=new ArrayList<String>();

ar.add("Hello");

ar.add("How");

ar.add("Are");

ar.add("You");

System.out.println(ar);

}

}