**LinkedList**

LinkedList is a class which is implementing List Interface as well as Deque Interface also .

It’s a very Special concept in term of pointer , in terms of references in java.

It is a dynamic Array which stored data on the basis of two things-

* One is Actual data and
* Reference data , ref of the next node

Example – I have a LinkedList and I have three Elements –

|  |
| --- |
| Head |

|  |  |
| --- | --- |
| Data | next |

Null value

|  |  |
| --- | --- |
| Data | next |

|  |  |
| --- | --- |
| Data | next |

1st Element 2nd Element 3rd Element

|  |
| --- |
| Head |

Null value

|  |  |
| --- | --- |
| 10 | next |

|  |  |
| --- | --- |
| 20 | next |

|  |  |
| --- | --- |
| 30 | next |

1st Element 2nd Element 3rd Element

This is called Singly LinkedList

--------------------------------------------------------------------------------------------------

| This is a node n1, n2 and n3| This is LinkedList having three different element.

In linkedList what exactly linkedList does ?

In LinkedList each and every node is created into two parts . And This particular node each and every node having two things .

1. The first thing is called Data |Whatever the data you want to put|
2. The 2nd thing is called next or references . Next means reference

Next—

**What do you mean by next?**

The next is the reference of next node. Next is pointing to next particular node. And each and every node is having data and next.

Q- What kind of element we can store?

Integer, String and different kind elements we can store.

Each and Every LinkedList having Two things ---

1. The thing is called Head |Head of the LinkedList| and
2. The 2nd thing is called

Head -It is not a node. It is simple starting point of that particular LinkedList. It is not divided into two parts . No data and No next. Head is pointing to First node data.

**Q Who is pointing to The next of Last Element ?**

The Last element next is pointing to nothing /Null. Because there is no other forth element is available . So last Next is referring to null.

Example – Let’s See If I want to put the data into it . Let’s see three data 10,20,30 and we can use simple Array –

ar [0] =10;

ar[1]=20;

ar[2 ]=30;

and In ArrayList we use dotadd();

ar.add(ar[2]);

Similarly these other three is there . In term of Java and Collection we called it Singly LinkedList.

Singly LinkedList -It means each and every node{ n1, n2 and n3} is divided into two parts . The first part stores a value and Second part stores the reference of the next element /node .It does not store the reference of the previous node.

Each and Every node is having the reference of the next element, that’s why this is called singly LinkedList.

Q -How to implement Singly LinkedList?

Ans – the following Code---

**package** NaveenCollections;

**import** java.util.Iterator;

**import** java.util.LinkedList;

**public** **class** LinkedListSinglyDemo {

**public** **static** **void** main(String[] args) {

LinkedList <String> lnk = **new** LinkedList<String>();

//add()

lnk.add("Test");

lnk.add("QTP");

lnk.add("Selenium");

lnk.add("Java");

lnk.add("FrameWork");

//Simple print

System.***out***.println("Contents of LinkedList - "+lnk);

//addFirst()

lnk.addFirst("Rehana");

lnk.addLast("Taseen");

System.***out***.println("Contents of LinkedList - "+lnk);

//get() - get value from index

System.***out***.println(lnk.get(0));

//Set() // Which value you want set newly with the other index.

lnk.set(4, "Automation");

System.***out***.println(lnk.get(4));

//remove first and Last element

lnk.removeFirst();

lnk.removeLast();

System.***out***.println("Contents of LinkedList - "+lnk);

//remove the specific element

lnk.remove(2);

System.***out***.println("Contents of LinkedList - "+lnk);

System.***out***.println();

//=========================

//Iterate/ print all the values of LinkedList

// for loop

System.***out***.println("\*\*\*\*\*\*Using for loop\*\*\*\*\* ");

**for** (**int** n = 0; n<lnk.size(); n++) {

System.***out***.println(lnk.get(n) );

}

//Advance for loop

System.***out***.println("\*\*\*\*\*\*Using advance for loop\*\*\*\*\* ");

**for**(String str : lnk) {

System.***out***.println(str); // srt string of all the LinkedList object

}

System.***out***.println("\*\*\*\*\*\*Using Iterator\*\*\*\*\* ");

//Iterator

// lnk.iterator() and this iterator will return Iterator reference

Iterator <String>itr =lnk.iterator();//here iterator name =itr.

**while**(itr.hasNext()) { // in this iterator the next element is there

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

}

System.***out***.println("\*\*\*\*\*\*Using while\*\*\*\*\* ");

//While loop

**int** num = 0;

**while**(lnk.size()>num) {

System.***out***.println(lnk.get(num));

num++;

}}}

/\*

**1) LinkedList is a class and create an object and I want to store/ add some String value 1st using add().**

**2) here I have defined the Generic also**

**3)Methods Of LinkedList----**

**i) add()**

**ii)addFirst()**

**iii)addLast()**

**iv) get() - from index**

**v) set()**

**vi) removeFirst()**

**vii) removeLast()**

**viii) remove() from a specific position**

**4) How to get inside/ insert the value?**

**Ans - Using get().**

**5) How to set inside the value?**

**Ans - using set(Index, element-what should be the element). After setting we have to get the setting value**

**then use get() with syso.**

**Q 6) How to Iterate/ print all the values of LinkedList?**

**ans --There are number of phases- 1) for loop 2) Advance for loop/ for each loop 3) Iterator 4) while loop**

**7) for(String str : lnk) {**

**System.out.println(str);**

**here srt string of all the LinkedList object. Whatever the string are available in this particular lnk.**

**please iterate one by one .Automatically [no need to define i++]**

**}**

**Q8 - There are different ways to print the values of LinkedList.How to do that?**

**Ans --There are number of phases- 1) for loop 2) Advance for loop/ for each loop 3) Iterator 4) while loop**

**\*/**

**-==============**

**OutPut**

**Contents of LinkedList - [Test, QTP, Selenium, Java, FrameWork]**

**Contents of LinkedList - [Rehana, Test, QTP, Selenium, Java, FrameWork, Taseen]**

**Rehana**

**Automation**

**Contents of LinkedList - [Test, QTP, Selenium, Automation, FrameWork]**

**Contents of LinkedList - [Test, QTP, Automation, FrameWork]**

**\*\*\*\*\*\*Using for loop\*\*\*\*\***

**Test**

**QTP**

**Automation**

**FrameWork**

**\*\*\*\*\*\*Using advance for loop\*\*\*\*\***

**Test**

**QTP**

**Automation**

**FrameWork**

**\*\*\*\*\*\*Using Iterator\*\*\*\*\***

**Test**

**QTP**

**Automation**

**FrameWork**

**\*\*\*\*\*\*Using while\*\*\*\*\***

**Test**

**QTP**

**Automation**

**FrameWork**

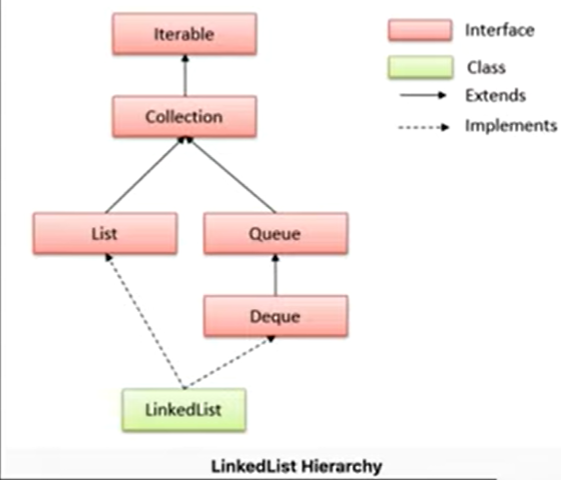
**Q What is linked list and what is the implementation internally? And How exactly linked lists work?**

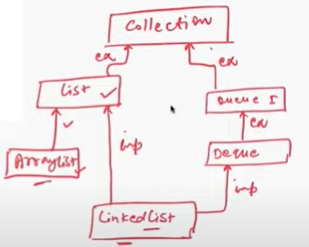
**Q How will you create your own linked list class** ?

The collection hierarchy that we have top collection interface.so this is the main connection and then we have a queue interface is available and then we have one more **deque interface** is available. and then

we have [this particular] linked list class is available and then this

particular linked list class is actually implementing deque interface. Deque is actually extending Queue interface and deque is finally extending this collection interface and then we have one list interface and this list interface is implemented by linked list also and then we have another implementation of list that is available in the form of Array List.





Now and this List is actually extending collection interface.

**what is the difference between ArrayList and the Linked list?**

Ans-

One of the different is that linked list is the implementation of

list interface. ArrayList is also implementation of list interface but

the another difference is that linked list will have the implementation of the queue and the deque also.

So whenever all the queue features also will be available, all the queue different methods also available inside the linked list and all the method which are coming from list interface also available in LinkedList. So the implementation of the list also available in linked list.

**What is linked list?**

There are two types of linked list-- one is the

* **singly linked list** and
* the second one is a **doubly linked list.** and

then this link list can be converted into **circular linked list** and

So let's see I want to store multiple values in this particular array

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10 | 20 | 30 | 40 | 50 |

-🡪 Array

So let's see this is 10, this is 20, this is 30 40 and 50 I have stored and

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1000 | 1004 | 1008 | 1012 | 1016 |

**Pic- Memory** Allocation

so it will be stored in the Contiguous [Contigious] Memory allocation.

So let's see this [10] is available on 1000 memory and it [20/ guys] will take four bytes, Let's see for integer to store this particular 20. so the next memory allocation will happen on 1004, 1008 and 1012 and 1016 over here like that. so it will take the contiguous memory allocation.

But the problem with this particular contiguous memory allocation is that. let's say I want to add one more values in between 20 and 30, let's see I have one value 15 I want to add it over here. so it means lot of shifting I have to do that it means I have to move all these 30 40 50 towards this side. Create one space over here [in between 20 and 30] and then I have to store 15 over here.

So the shifting is very much needed over here and in that case

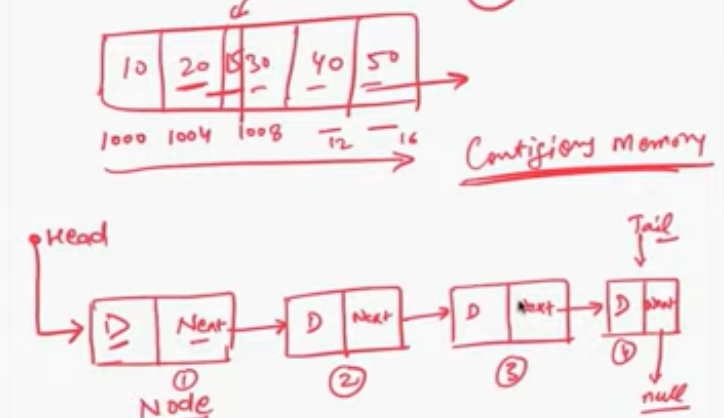
if you're doing this particular operation multiple times .so this is not the right solution. And this is not a feasible solution for that. So every time shifting I have to do that, either I have to shift the entire array to the left side or to the right side over here.

So this is the problem with the contiguous memory allocation.let's say I want to store all the employee id in the sorted order.

**Single link list** is nothing just maintains different nodes over here

like that.





So let's see this is node number one, this is my node number, two this is my node number three, and so on.

and every node is actually having two parts.The first part is for data second part is the pointer, that I would say next pointer which is pointing to the next node over .

Same data 🡪pointing to the next node over here which is pointing to this data pointing to the next over here. pointing to another node and data and pointing to next over here and this last node is pointing to null and

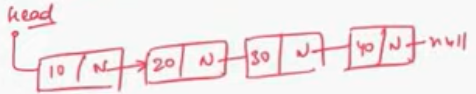
this is the head is available and this is called head the first node

pointed by a head. So this is the head of the linked list and this is called the tail of the linked list. and every node this is called a

node over here.

This is node number one , this is node number two, this is node number three and this is node number four. every node is having two sections one is the data and then the next pointer to the next node and I can store the data in this particular forward direction only.I cannot store the data in the backward direction. it means I cannot come over here just before that particular linked list. So the data will be stored in the next Direction. This is called singly linked list.

if I really want to store if 10 20 30 40.so I will create let's see a 10 over here and the pointer will be pointing to the next node where the value is 20, , this pointer is pointing to the next node value is 30 and this pointer is pointing to the 40 over here and now this is pointing to a null and this is the head like that.so this is my next pointer I would say like that. so this is called a singly linked list.



This particular node createa nodes and then create a linked list

**Create nodes and then create a linked list.**

We are going to create a node class

class Node {

and then every node is representing this.let's see this particular box is represented by a node. so every node will have a data first of all

whatever the data that we have**—int data I;** and then every node will have the pointer that is called next pointer which is type of the node type—**Node next;** and then create the constructor of this particular node class–**Node ()**and there i'm passing that okay hey what type of data you want. so let's say i'm passing integer type of data integer data type of data.

**Node (int data) { // constructor of the Node class**

**this.data=data;**

**}**

and then whatever the next node that you are creating which is pointing to null.—**next = null**

I just want this node class is used to create the nodes. so this node class will create a node like that having two things one is the data and the next is the pointer. so data will be stored over here  and next one is the pointer over here.Next point is always pointing to null and this is the head. So this is I have added one node over here

so this is called one node linked list.



**class** Node {

**int** data ;

Node next;

Node(){ // constructor

**this**.data = data;

next = **null**;

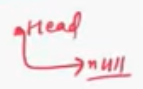
}

}

If I want to create empty linked list it means the head. this

Head is actually pointing to nothing .it's actually pointing to null.

it means I don't have any notes in this particular link list that is called empty link list.



**Q—How and What is implementation of linked list?**

Linkedlist is a class which is already available in java.

Create your own LinkedList ----

when we are designing a system or let's see in any production based system, in that case obviously we will be using the existing linked list API or linked list class.

At the time of interview –

**Can you create your own linked list ?**

Steps –

1. Go to Eclipse
2. Create a Java class with main method.
3. Create a class here. Name is Node over here.

In the Node class write int data; [what kind of data you want to store] so this class is the inner class of this particular Linked list class.

1. Create a Node which is point to the next pointer
2. Create a constructor of this particular node
3. Write give me the data [int data- in the constructor]and the same data

I am giving it to- this.data =data;

1. whatever the next pointer which is pointing to null.

So simple a node class got created over here like that

**public** **class** MyLinkedList {

**class** Node{

**int** data;

Node next;

Node(**int** data){

**this**.data=data;

}}

1. ) Now inside the main method I am going to create number of nodes.
2. create the object of my linked list class-
3. create a node over here using ref variable of this object variable. In this the constructor will be called let’s I am passing 10 over here.

MyLinkedList ll = **new** MyLinkedList();

ll.**new** Node(10);

1. now I have designed this particular node only for integer data. you can create for string also or for any other data type. if you want to use that you can do that.
2. so ll.node I have written and then I'm creating the node and I'm giving the node number first.

Node first = ll.**new** Node(10);

**13)so node first is equal to ll.new node() and then because this is the inner class and and this is not the static class, so object I have to access by the object reference name. ll.**new node and I'm passing 10 over here. so 10 will be given to **this** guy [Node(**int** **data**)] and 10 will be given to the class variable [**int** data;]with the help of ‘this’ keyword.

|  |  |
| --- | --- |
| 10 | n |

so this node got created like that and the value is 10 and this is my next pointer denoted by let's see N or next.

So one node got created then I have to create a head of this particular node. so at this particular linked list class label I'm going to create one head at the class level which is the node type head and then I have to create a head. so head will be pointing to this –

|  |  |
| --- | --- |
| 10 | n |

So you are writing ll.head= first; How to access head I simply write ll.head, because this is my class variable, after access by the object reference name is equal to first. ll.head= first;

so this linking is done. It means I have created a node and I have created this head pointer which is pointing to my first node over here.

then I'm going to create one more node. so let's see i'm going to create one more node it means the moment I create this particular node, this node got created and the value is a 20 and the pointer is next. So this is my first node, this is my second node. so

I have to do first.next = second ;

so that the next node of the first node will be connected to the second node over here. so I have to do this particular mapping or linking. So to connect or Linking I simply write ---

---go to my first node.next= second; so the linking is done now the first dot node is equal to second node same way i'm going to create

one more node.

**Print--**

I want to print the value from these particular nodes. So I'm going to create a method over here. Let's see simple create a method

**public** **void** printMyLinkedList()

{

}

To print it, create a reference of Node n is equal to which is pointing to head .

**why pointing to head?** Because I have to start my counting from head only. so this is my head and this is my tail.

so I have to keep traversing towards this particular direction first node, second node, third node and I have to keep checking that the next pointer of any node is pointing to null. so when I find the next pointer is equal to null then I have to stop my counting then I have to stop my traversing over here. And That’s why I simply create a **while loop** over here and

then I simply write while n is not equal to null [**while** (n!= **null**) { }]. you keep running until you find null over there and then System.***out***.println();print value (n.data) ,it means n is going to first and what is the value?--> data. data is 10 will be picked and it will print 10 over here and then I have to move my counter to the next position. so I simply write n=n.next; over here. So n is pointing to n dot next again we'll check , n is not equal to null🡪. no n is pointing to 20 this time it will pick the data and then 20 will be printed on the console and then again and not next it will point into 30 the next of this is 30. 30 will be printed on the console and then n not next will become null. Because this next is pointing to null. So n will become null. The condition is not satisfied over here and then it will come out of the while loop . then call this particular method

in the main method. So my object name is ll dot print my linked list🡪 **ll.printMyLinkedList(); .So the result will be shown in the console like this—**

10

20

30

40

50

**Conclusion of LinkedList ---**

So this is how I **have** created.

* I have created my own linked list
* I'm not using linked list class which is provided by java by default.
* I'm creating my own linked list implementation with the help of

integer data with the help of nodes over here.

**package** com.qa.linkedlist;

**public** **class** MyLinkedList {

Node head;

**class** Node{ //Constructor of Inner class

**int** data;

Node next;

Node(**int** data){ // Constructor of Node class

**this**.data=data;

}

}

**public** **void** printMyLinkedList()

{

Node n = head;

**while** (n!= **null**) {

System.***out***.println(n.data); // Print n data

n=n.next; // print serially n's data

}

}

**public** **static** **void** main(String[] args) {

MyLinkedList ll = **new** MyLinkedList();

Node first = ll.**new** Node(10);

ll.head= first;

Node second = ll.**new** Node(20);

first.next= second;

Node third = ll.**new** Node(30);

second.next = third;

Node fourth = ll.**new** Node(40);

third.next = fourth;

Node fifth = ll.**new** Node(50);

fourth.next = fifth;

ll.printMyLinkedList();

} }

We have to understand that this is the manual mapping I'm doing all

the time what if tomorrow you have 50 data or 10 data or like that

are you going to map the data like that no so we will be creating one add function or add method and then we will keep passing the node over there and then we will keep connecting the nodes to each other fine so that's all for today I hope you got it very famous interview question

please write this code by your own and people will ask this question at a time of interview that

How will you create your own linked list? so you have

* to create a Inner class a node class
* Create a method to print, while loop
* create a method and do not forget to do the mapping –

Node first = ll.**new** Node(10);

ll.head= first;

Node second = ll.**new** Node(20);

first.next= second;

Node third = ll.**new** Node(30);

second.next = third;

First individual nodes will be created and then you have to do the mapping this arrow is called mapping. So you have to write head is pointing to first the first neck is pointing to second and second next pointer is pointing to third and so on and

we just need to print it with the help of a while loop until you find the next pointer is equal to null.

**Questions** –

1. What is linked list and how exactly it works internally

So we have created one node and we have implemented our own link.

**That's it a very simple logic and almost every company either you go to amazon flipkart or microsoft everywhere people are asking such kind of questions.**

**19-LinkedList various features and methods (Important for Interviews)**

**Questions –**

1. Different methods and Features in linked list.
2. How exactly linked list works internally.
3. Linked list is a default class which is already available in java. So you don't need to implement linked list all the time.
4. along with list interface it implements a queue and deque also.
5. it can be used as list and the for a stack and you can use it with the queue also as it implements all these interfaces.
6. so this is the advantage of linked list as compared to Arraylist because it's having the feature of list as well as stacking

**Steps to be done**

1. Create a class with main method that is linked list methods class.
2. Create couple of important documentation and couple of Nodes

it allows the null entry. We can have null values over there and it is a dynamic collection.

it allocates memory whenever it's required therefore insertion and deletion operations can be easily implemented.

So it is dynamic collection hence I will write a basic node over here that insertion and deletion can be write easily.

it can contain duplicate elements so it can contain duplicate elements and

It is not synchronized or thread safe.it means multiple threads can access can do the read and write operation at the same time.

Manipulation is fast as compared to arrays because we don't need any shifting base because it contains a pointer value which is pointing to the next node. so that's why we don't need any kind of shifting.

so if you take a typical linked list let's see this is as in the previous example we have seen that that we have data and this is the next pointer.

|  |  |
| --- | --- |
| 10 | n |

|  |  |
| --- | --- |
| 20 | n |

next node having the data next pointer, next node having the data and the next pointer over here. So this next pointer is actually connected to this data. this next node is connected to the third node data.

So the shifting is very easy. we don't need to do manipulation and we don't need to do

any kind of shifting over here. I want to introduce one more node between these two I



break the connection from here next point will point to start this and the next is start pointing to this that's it .

but in case of arrays, this is a typical static area that I have it. in this particular static area the values are already available .let's see some values are already available over here. a new value that I want to store over here. I have to do a shifting over here. This entire array from this particular position 2 to n number of position have to shift towards this side and then and then this particular value will be added over here like that. So that's why these kind of operations are very expensive as compared to linked list. So in this case I would always prefer linked list instead of a typical static array.



**Constructor-**

Linked List talk about **Constructor.**ifferent **Constructor** are available in LinkedList. There are two basic constructors ---

* **one is a typical linked list constructor which is a default constructor.**
* **second linked list constructor is you can create with collection with the existing collections.**
* **//LinkedList Constructor –**

**1) LinkedList (); constructor and**

**2) LinkedList(Collection c);**

These are the two constructors are available.

There are various methods that we collected a list from the java documentation of the jdk –

add method(), add all method () and add first(), clear (),cloning (),element get() index of(), last index(), iterator ()method, remove method, polling methods etc.

**Q- How to declare one LinkedList?**

**LinkedList**

**linked list** is the class and we need to create the object of linked list and

let's say I want to create a linked list with the string values <>.so as a generics I have to give a string over here.

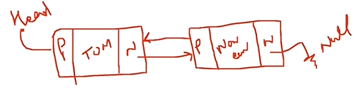
And Like LinkedList name is ‘names’ and then import this particular link list from java.util package..

Now from this particular names I'm going to add there is a add method add and what kind of values I can store. let's see Ruma. so one value got added in this particular node.

Our node will be created in this particular node. Ruma will be added and this node is pointing to null, the next pointer is pointing to null. So whenever I'm writing this particular line [LinkedList <String> names = **new** LinkedList();] Initializing my linked list and

Immediately I want to check that hey what is the size of this particular list?

so I simply write --names.size();



so if you run it let's see what is the size. so run as a java application it's saying zero days it's saying zero over here. Because now we don't have any element my physical capacity of this particular linked list is zero but now I'm going to add one value over here and one more value let's see I'm going to add tom and I'm going to add Ruma and then I'm going to check the size once again so it will give you 2 over here and let's run it and let's see so you can see who got added right and then I'm running in debug mode and I'll show you one interesting thing over here put a debugger at light number 27 and right click on it go to debug as java application and uh let's see what happens

so yes at line number 27 it will stop and you can just mouse over on names and you see that this is the see internal implementation we will see that okay how exactly

it is maintaining the data see this is the first pointer and the last pointer can you see that if you open this particular first pointer the value is tom and its next

pointer is equal to null all right guys and its previous pointer is also equal to null and then you open this particular last pointer that is item is equal to uh null

and its next pointer is pointing to done it means you can see that okay tom got

added over here right in this particular first node you can see that now what exactly I'm going to do I'm going to add navi now so let's add Ruma and let's see tom and Ruma both got added and now you open this particular first pointer see it carefully how exactly it is implemented and then see this the first pointer item is equal to tom and next pointer is actually pointing to Ruma can you see that it's next pointer so this node next pointer is pointing to Ruma and the previous node is

pointing to null because this is the head and then again the last pointer that we

have added is Ruma over here and it's previously pointing to tom over here so it is creating a doubly linked list internally over here like that

**So if you see this in the form of a basic diagram how many nodes got created two nodes node number one and node number two like that and it's actually divided into three parts let's see** I'm writing it like that one and let's see two and this is the data first data is

our tom over here and the second data I have added again that's why I really

=============================================================================

**Q-How will you create your own LinkedList Class ad Create your own method in this particular link list?**

1. **Create your own size ()**

**Q-How exactly internally they are implemented and how will you create your**

**own custom linked list**?

package com.qa.linkedlist;

import java.util.Collections;

import java.util.Iterator;

import java.util.LinkedList;

import javax.sound.sampled.AudioFileFormat.Type;

public class LinkedListMethod {

public static void main(String[] args) {

//It is default class in Java.

// It can be used as List, Stack, Queue

// It allows null entry

//Dynamic collection , hence insertion and deletion can be easily implements

// It can contain duplicate elements

// It is not synchronized or thread safe.

// In LinkedList, manipulation is fast because we don't need any shifting

//LinkedList Constructor---1) LinkedList (); and 2) LinkedList(Collection c);

//====================================================

// How to declare one LinkedList?

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

System.out.println(names.size()); // here is no Size. ie. size =0

// This is the overloaded method

names.add("Ruma"); // 0 th index

names.add("Afsheen"); // 1 st

names.add("Taseen"); // 2 nd

names.add("Kadir"); // 3rd

names.add("shaheen"); // 4th

System.out.println(names.size()); // here size 3 becausee three data here

System.out.println(names); // Directly print over here.

// In Zeroth index what exact value is available

System.out.println(names.get(4)); //Result --shaheen.

//here total index -Ruma(0 index),Afsheen (1 index), Taseen (2nd index.).So 3rd index is not available

//But if I try to get 3rd index then we get IndexOutOfBoundsException . so you should not go beyond the index.

//so i cannot take like that

//=========================================================

// how to Iterate/ how to iterate [use names.iterate () . We have to apply Iterator methods]

// Iterator will give you -- the Iterator of String

// I want to print all the values from this particular ArrayList

Iterator<String> it =names.iterator();

while(it.hasNext())

{

System.out.println(it.next()); // printing the value and it will print all the values

}

//=====================================================

// there are 4th index we have . So I simply say that on the fourth index, let's add some other

// guys.so let's see I'm writing 'samir' over here. and then I'm going to print a names once again over here.

// so let's see what happens. So let me delete this particular drawing and let's run it again

names.add(5, "samir");;

System.out.println(names);

System.out.println("================================");

// On the second index I want to add new value over the second index.I.e in the 2nd index

//Taseen will be replaced by 'nura'. That means Taseen will be shifted. Replace means it will be shifted over here.

// not exactly replace

names.add(2, "nura");;

System.out.println(names);

System.out.println("==============X================");

/\*The shifting is very easy. Tthe moment Afsheen will be Afsheen's next node

will be connected to Taseen and Taseen next node will be connected to Kadir like that.

So it will create a doubly linked list like this .so Ruma is connected to Afsheen . Afsheen is connected to

Taseen. Taseen . So it will create a list over here like that.

so initially between connection like that between Afsheen and Taseen . Now we have to add nura.

So now Afsheen will start pointing to nura and nura will start pointing to taseen. that means taseen will be shifted.

\*/

//===========================================

/\* Let's see create a new list and this is my users list. and in this particular

users list I'm going to add a couple of users.let's see users.add and

now the new list I'm adding 'peter' and "Trump". So I have created this new list and

I want to add with the oldest LinkedList 'names' list.

So Simply write names.addAll() method is available and

you can pass the existing connection that you have already created this..> |names.addAll(users);|.

and then print names. \*/

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

users.add("Peter");

users.add("Trump");

names.addAll(users);

System.out.println(names);

names.addFirst("Karim");

System.out.println(names);

names.addLast("devandi");

System.out.println(names);

names.remove(3);

System.out.println(names);

names.removeAll(users );

System.out.println(names);

names.removeFirst();

names.removeLast();

System.out.println(names);

//From the names list

//remove all the elements which are coming from the users list.

System.out.println("==============================");

names.clear();

System.out.println(names);

System.out.println("===========================");

// create one more Linked List.

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

System.out.println(lang.size()); // Initally size = 0

lang.add("JAVA");

lang.add("Python");

lang.add("Rubi");

lang.add("JavaScript");

System.out.println(lang.size());

//Reverse the LinkedList: [with the help of Iterator ]..Do the traversing

Iterator<String> itr = lang.descendingIterator();

while(itr.hasNext()) {

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

}

System.out.println("====================");

//You can do the traversing in a normal way also with the help of for each loop

//1) type for and what type of value we have --> String. then create any variable 'e' and

// and then what is the list name--> lang[language] and then print.

// Then it will start printing from java to javascript.

for(String e : lang) {

System.out.println(e);

}

//

//Collecting Sorting - how you will sort a Particular ArrayList-With the help of Collections class.

// there is a method--> sort ().and pass language -lang over here and then print language list.

//sort :

System.out.println("===================");

Collections.sort(lang);

System.out.println(lang);

// So it is coming in the sorted order.=======/[JAVA, JavaScript, Python, Rubi]

}}

**Output in the console**

/\*

0

5

[Ruma, Afsheen, Taseen, Kadir, shaheen]

shaheen

Ruma

Afsheen

Taseen

Kadir

shaheen

[Ruma, Afsheen, Taseen, Kadir, shaheen, samir]

================================

[Ruma, Afsheen, nura, Taseen, Kadir, shaheen, samir]

==============X================

[Ruma, Afsheen, nura, Taseen, Kadir, shaheen, samir, Peter, Trump]

[Karim, Ruma, Afsheen, nura, Taseen, Kadir, shaheen, samir, Peter, Trump]

[Karim, Ruma, Afsheen, nura, Taseen, Kadir, shaheen, samir, Peter, Trump, devandi]

[Karim, Ruma, Afsheen, Taseen, Kadir, shaheen, samir, Peter, Trump, devandi]

[Karim, Ruma, Afsheen, Taseen, Kadir, shaheen, samir, devandi]

[Ruma, Afsheen, Taseen, Kadir, shaheen, samir]

==============================

[]

===========================

0

4

JavaScript

Rubi

Python

JAVA

====================

JAVA

Python

Rubi

JavaScript \* /

=============================================================================