Collections Q/A

**Q. Arrays vs ArrayList ?**

Ans :

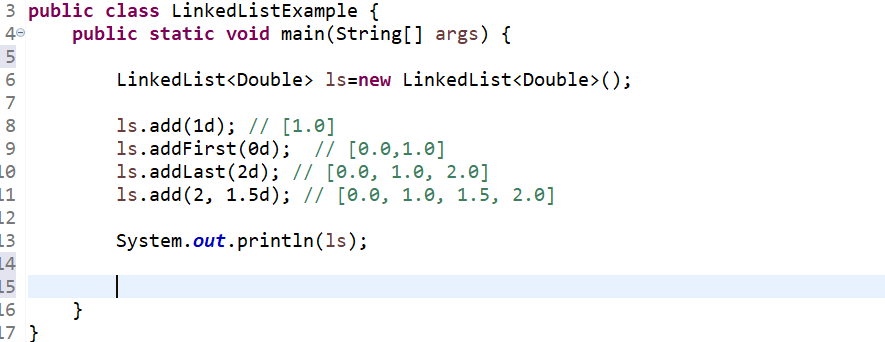
|  |  |
| --- | --- |
| Arrays | ArrayList |
| Arrays size is fixed and need to provide while created | ArrayList dynamic no need to provide while creating |
| Array support primitive and object , does not support generics. | ArrayList support object and generic , does not support primitive data type. |
| In Array you assign values to index | Where in ArrayList to add value at the end of list |
| To retive size of Array we use “.length” property | Where in ArrayList we need to call “.size()” method |

Q.21 what is difference between ArrayList vs LinkedList ?

Ans :

|  |  |
| --- | --- |
| ArrayList | Linked List |
| Internally it implements Array. | Interanally it implements Double Linked List |
| It is slow in manupuation | Fast in manupulation |
| It is fast in search | Slow in search |
| Only treat as List | You can use this as list,queue,stack |

Example of linked list



**Q. Working of hash Map ? (https://www.youtube.com/watch?v=SXfsBDTodpY)**

Ans :

HashMap create segment of 16 index once it created ( 0-15 )

Map<String,Integer> mp=new HashMap<>();

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

Put operation :

Step 1 : calculate hashcode of key

Step 2 : calculate index from hashcode [ index = hashcode & (n-1 ) ]

Step3 : create node .

|  |  |  |  |
| --- | --- | --- | --- |
| Key | Value | Hashcode | Reference of next node |

Note : hash code of null will be zero .

Get Operation :

Step 1 : calculate hashcode of key

Step 2 : calculate index from hashcode [ index = hashcode & (n-1 ) ]

Step 3 . Goes to that index check hashcode

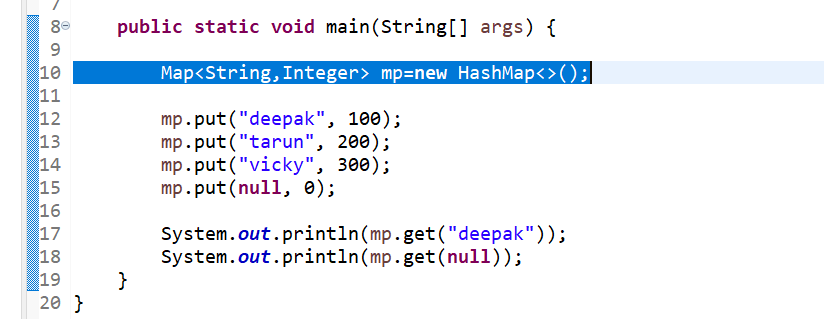
step 4.A : if hashcode is same than compare keys using equals method if equals method return

true than it return value .

OR

step 4.B

if hashcode is same but keys are not or equals return false then move to next element in the list.



Line number 10 : create segment of 16 index .(0-15)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

**Line number 12 :**

Step 1 . create hash code of “Deepak” which is key .

For Exp. Deepak hashcode is “112233”

Step 2. Calculate index from hash code of “Deepak” which “112233”

Index = 112233 & 15; [ hashcode & (n-1);

Index = 4;

|  |  |  |  |
| --- | --- | --- | --- |
| Key = deepak | Value=100 | Hashcode = 112233 | Ref of next node =null |

Step 3 : create a node

Step 4 : store this node ad 4th index.

**Line number 13: (when hashcode differ but same index or same hash code we will follow below mention step ).**

Step 1 . create hash code of “tarun” which is key .

Exp. tarun hashcode is “112244”

Step 2. Calculate index from hash code of “tarun” which “112244”

Index = 112244 & 15; [ hashcode & (n-1);

Index = 4;

|  |  |  |  |
| --- | --- | --- | --- |
| Key = tarun | Value=200 | Hashcode = 112244 | Ref of next node =null |

Step 3 : create a node

Step 4 :

Now we know that we already have value at 4th index. We add the reference of this node in reference column of already present node at 4th index like a chain as a linkedlist

**Line 17 :**

**Step 1 :**

Step 1 . create hash code of “Deepak” which is key .

For Exp. Deepak hashcode is “112233”

Step 2. Calculate index from hash code of “Deepak” which “112233”

Index = 112233 & 15; [ hashcode & (n-1);

Index = 4;

Step 3 . at index 4 .

3.1 : compare hashcode.

3.2 A.:

then check keys with equals method if return true. Then return value .

or

3.2 .B. if equals method return false . then go to next node and check.

**When Key is null in hash Map ?**

**Ans :** Element store at zero index , null hashcode is zero .

**Improvement in hashmap java 8 ?**

**Ans :**

We know that multiple object can have same hashcode or different hashcode but same index . then java 7 create a linked list. In case worst scenario **o(n)** . which can create performance impact.

But in java 8 when list reach a threashold limit which is “8” then list automatically convert into “balanced tree” . which is having time complexity of **O(Log n)**.