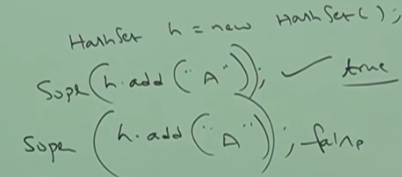
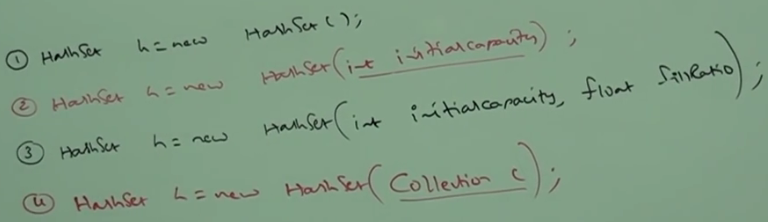
We have completed List part. 🡪 ArrayList, LinkedList, Vector, Stack.

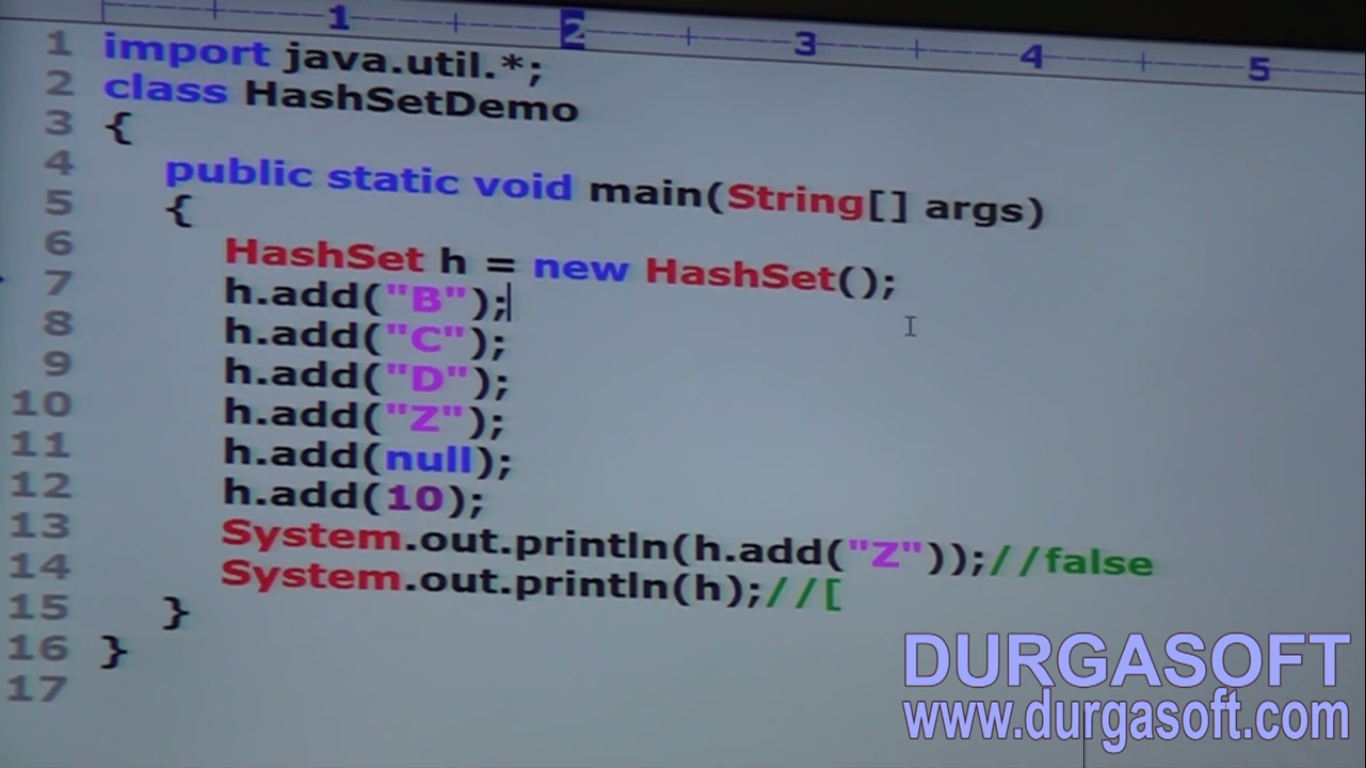
Set (I)

1. Set is child interface of Collection (I).
2. **Application**: If we want to represent a **group of individual objects** as a **single entity** where duplicates are not allowed and insertion order is not preserved.
3. Set interface doesn’t contain its own methods but coming from Collection(I)

HashSet(C)

1. **Properties**:
   1. **Underlying Data Structure:** Hash Table
   2. **Duplicate**: Allowed
   3. **Insertion Order**: Not preserved. As object is inserted based on Hash Code of the object.
   4. **Null Insertion:** Only Once. As duplicate is not allowed.
   5. **Heterogenous objects**: Allowed.
   6. **Implements**: Serializable, Cloneable, but not RandomAccess
   7. **Thread-Safety**: No
   8. **Best For**: When Frequent Operation is “search”
2. **NOTE:** In HashSet, duplicates are not allowed. If we are trying duplicates then we will not get any compile time or runtime error and add() simply returns **false**.   
   
3. **Constructors**:
   1. **HashSet()**:  
       Creates an empty HashSet object with   
       **default initial Capacity** = 16  
       **Default Fill Ratio**: 75% = .75
   2. **HashSet(int intialCapacity)**:  
       Creates an empty HashSet object with   
       **initial Capacity** = specified in constructor  
       **Default Fill Ratio**: 75% = .75
   3. **HashSet(int initialCapacity, float fillRatio)** Creates a new HashSet with specified initial capacity and specified fill Ratio.
   4. **HashSet(Collection)**:   
       Creates an equivalent HashSet object for the given Collection. This constructor is meant for inter-conversion for collection objects.



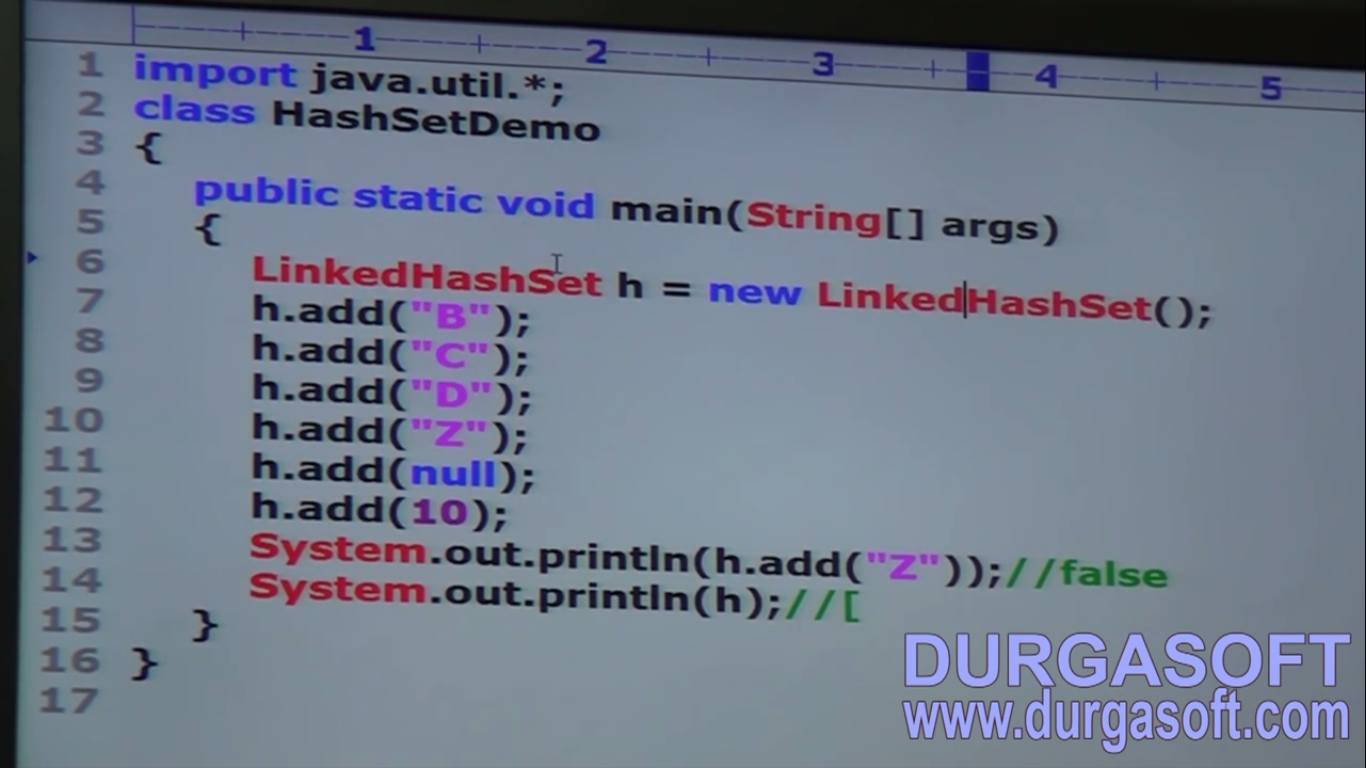
1. **Fill Ration or Load Factor:**
   1. After filling how much ratio, a new HashSet object will be created. This ratio is called 🡪 Fill Ratio or Load Factor.
   2. **Example**: 0.75 means after filling 75% ratio, a new HashSet object will be created.
2. **Example:**
   1. 
3. d

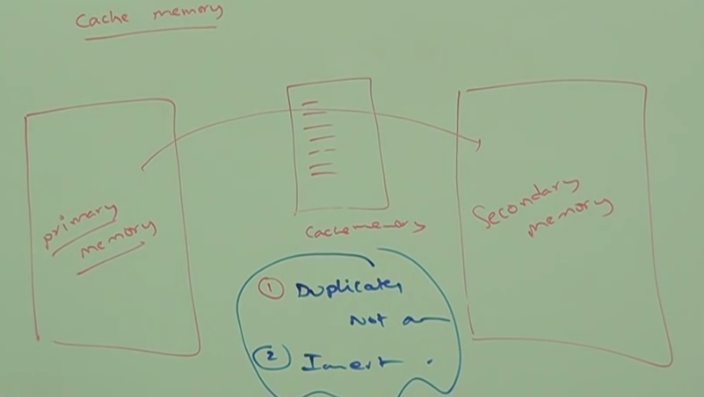
LinkedHashSet(C)

1. It’s the child class of HashSet.
2. It’s **exactly same** as HashSet (**including constructors and methods**) **except the** following differences

|  |  |
| --- | --- |
| **HashSet** | **LinkedHashSet** |
| **Underlying DS**: Hash Table | Hash Table, Linked List |
| **Insertion order:** Not Preserved | Preserved |
| **Version**: 1.2v | 1.4v |

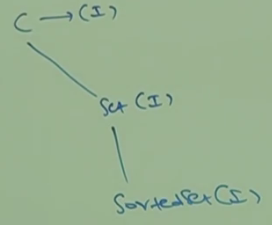
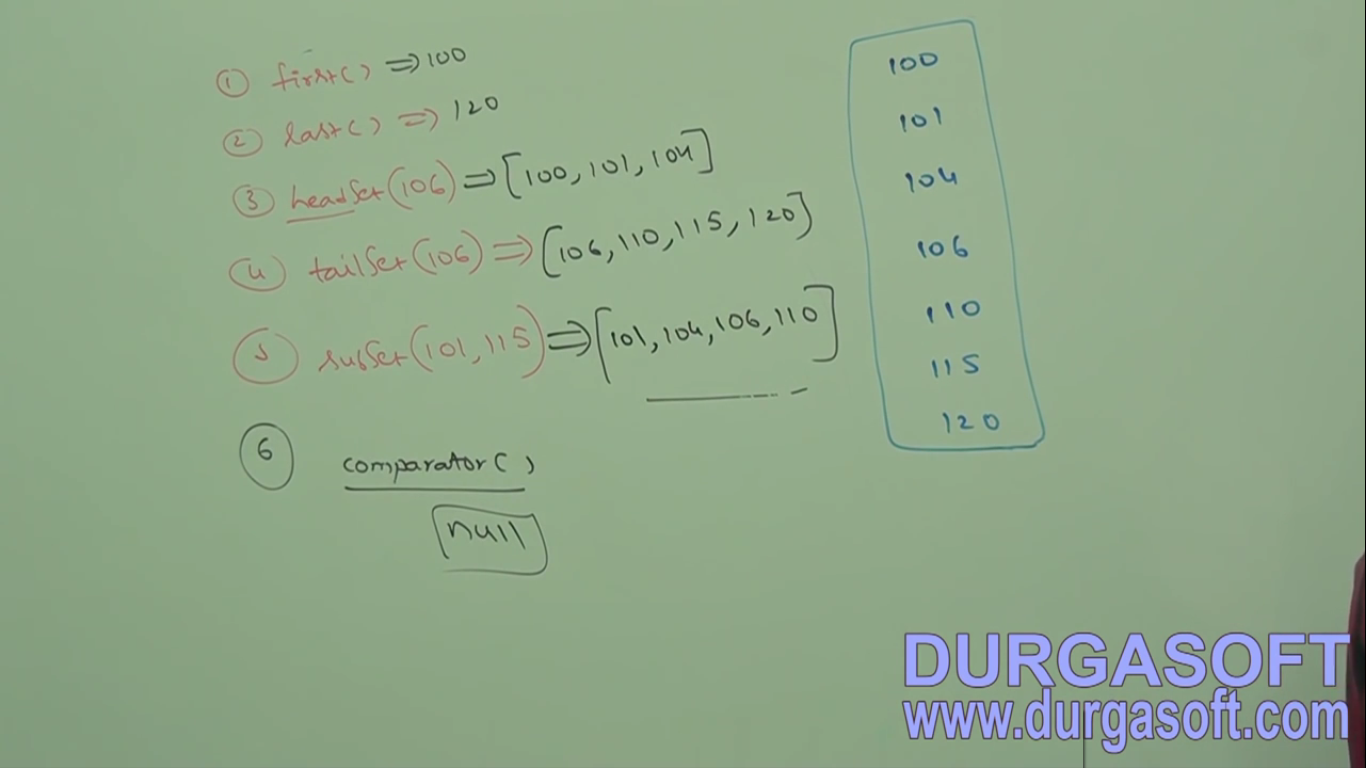
1. **Example**:



1. **Application**:
   1. To use cache Based applications.  
      

Insertion order must be preserved.

SortedSet(C)

1. SortedSet is the child interface (I) of Set (I). If we want to represent a group of individual objects according to some sorting order without duplicates.   
   
2. **SortedSet Specific Methods**:
   1. 
3. d
4. d