

## List

List is an interface under List. They are two types those are

1) Array list (C)

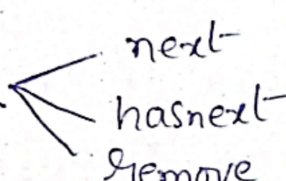
2) Linked list (C)

The loops used in the list are

1) for

2) for each

3) Java-8

4) Iterator.   
 

List using ArrayList :-

→ Java ArrayList class can contain duplicate elements

→ Java ArrayList class maintains insertion order

→ Java ArrayList class is non synchronized.

→ Index is maintained.

Syntax : ArrayList < Integer > al .

= new ArrayList < Integer > ( );

1) ArrayList:

```
public class ArrayListExample {  
    public static void main (String[] Args) {  
        ArrayList < String > list = new ArrayList < String >  
            ();  
        list.add ("Mango");  
        list.add ("Banana");  
        list.add ("Cherry");  
        list.add ("Jackfruit");  
        System.out.println (list);  
    }  
}
```

output: Mango, Banana, Cherry, Jackfruit.

2) ArrayList using Iterator: —

```
public class ArrayListExample {  
    public static void main (String Args[]) {  
        ArrayList < String > list = new ArrayList < String > ();  
        list.add ("Mango");  
        list.add ("Banana");  
        list.add ("Cherry");  
        list.add ("Jackfruit");  
        Iterator < String > itr = list.iterator ();  
        while (itr.hasNext ()) {  
            System.out.println (itr.next ());  
        }  
    }  
}
```

3) ArrayList using for each loop: -

```
public class ArrayList-Example 2 {  
    public static void main (String[] args)  
    {  
        ArrayList<String> list = new ArrayList<String>(  
            );  
        list.add ("mango")  
        list.add ("Banana")  
        list.add ("cherry")  
        list.add ("Jackfruit")  
  
        for (int i = 0; i < list.size(); i++) {  
            System.out.println(array list.get(i));  
        }  
    }  
}
```

output:- Mango, Banana, cherry, Jackfruit

4) Array list using for each loop: -

```
public class ArrayList-Example 3 {  
    public static void main (String[] args) {  
        ArrayList<String> (list) = new ArrayList<String>(  
            );  
    }  
}
```



```

list.add ("mango");
list.add ("Banana");
list.add ("cherry");
list.add ("Jack fruit");
for (String i: list) {
    System.out.println(i);
}
}
}

```

5) list using linked list :-

- Allow duplicates
- Allow Null values
- Allow Insertion order.

5) Program :-

```

public class Tyo {
    public static void main (String[] Args) {
        LinkedList <String> list = new LinkedList
        <String> ( );
        list.add ("Tyo");
        list.add ("mouni");
        list.add ("Rachana");
        System.out.println (list);
    }
}

```

6) using for loop: -

```
public class Tjo {  
    public static void main (String[] args)  
    {  
        LinkedList<String> l = new LinkedList<String>  
            ();  
        1. l.add("Tjo");  
        2. l.add("mouni");  
        3. l.add("Rachana");  
        for (int i = 0 ; i < l.size(); i++)  
        {  
            System.out.println ( );  
        }  
    }  
}
```

7) using for each loop: -

```
public class Tjo {  
    public static void main (String[] args)  
    {  
        LinkedList<String> l = new LinkedList<String>();  
        l.add("Tjo");  
        l.add("mouni");  
        l.add("Rachana");  
    }  
}
```

~~1) Each system.out.print ( );~~

```
for (String str : l)
```

```
    System.out.print (str + " ");
```

```
}
```

```
}
```

2) Using Iterator :-

```
public class Tjo {
```

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

```
        LinkedList<String> l = new LinkedList<String>();
```

```
        l.add ("Tjo");
```

```
        l.add ("Mouni");
```

```
        l.add ("Rachana");
```

```
        Iterator<String> iterato = l.iterator();
```

```
        while (iterato.hasNext()) {
```

```
            System.out.print (iterato.next() + " ");
```

```
        }
```

```
    }
```

```
}
```



Set: - set is an inter under set

They are : 1) Hash set (C) → Linked hash set

2) Sorted set (C) → TreeSet

HashSet: -

1) Java for each loop

```
public class {  
    public static void main(String[] args) {  
        Set <String> l = new HashSet <> ();  
        l.add ("Java");  
        l.add ("python");  
        l.add ("spring boot");  
System.out.println ("see: d. ita");  
        for (String language : languages l) {  
            System.out.println (language);  
        }  
    }  
}
```

Iterator:-

```
public class {  
    public static void main(String[] args) {  
        Set<Integer> number = new HashSet<>();  
        number.add(12);  
        number.add(13);  
        number.add(14);  
        Iterator<Integer> iterate = number.iterator();  
        System.out.println("Iterating over set:");  
        while (iterate.hasNext()) {  
            System.out.print(iterate.next() + ",");  
        }  
    }  
}
```

1) foreach :- public class {

```
    public static void main(String[] args) {  
        Set<String> s = new HashSet<>();  
        s.add("Syo");  
        s.add("mouni");  
        s.add("Nikhil");  
        s.forEach(System.out::println);  
    }  
}
```



linked Hash set: - (for-each)

(12)

```
public class Jyo {  
    public static void main (String[] args) {  
        LinkedHashSet <String> jyo  
            = new LinkedHashSet<String>();  
        jyo.add ("caumi");  
        jyo.add ("Naidu");  
        jyo.add ("Hima");  
        for (String itr : jyo) {  
            System.out.println(itr);  
        }  
    }  
}
```

(13)

Iterator: - Iterating Linked Hashset -

```
public class Jyo {  
    public static void main (String[] args) {  
        LinkedHashSet<String> s = new LinkedHashSet<  
            <String>();  
        s.add ("caumi");  
        s.add ("Naidu");  
        s.add ("Hima");  
        Iterator itr = s.iterator();  
        while (itr.hasNext()) {  
            System.out.println(itr.next());  
        }  
    }  
}
```

Java-8 (for each)

(14)

```
public class Tjo {  
    public static void main (String [] args) {  
        Linked HashSet <String> s = new HashSet<String>  
            ();  
        s.add ("Caemi");  
        s.add ("Naidu");  
        s.add ("Hima");  
        s.stream ().forEach (System.out::println);  
    }  
}
```

(15) Sorted Set :-

(Iterator) :-

```
public class SortedSet {  
    public static void main (String [] args) {  
        SortedSet <String> set = new TreeSet<String> ();  
        set.add ("welcome");  
        set.add ("To");  
        set.add ("parvathipuram");  
        set.add ("5");  
        System.out.println ("Sorted Set: " + set);  
        Iterator iterator = set.iterator();  
        System.out.println ("iterator");  
    }  
}
```



```

while (value.hasNext()) {
    System.out.println(value.next());
}
}
}
}

```

map: - map is an interface

It can be classified into two types.

1. <sup>HashMap</sup>Linked Hash map (C)
2. Sorted map (C) → TreeMap
3. Hash Table (C)
4. Concurrency Hash map (C)

(16) Hash map using EntrySet: -

```

class Iteration (HashMap)
{
    public static void main (String[] args)
    {
        Set<Entry> entrySet = map.entrySet();
        Map<String, String> v = new HashMap<
            String, String>();
        v.put ("Jyothsna" "panvathipuram");
        v.put ("Mouni" "Devarapalli");
        v.put ("Rachana" "Narsipatnam");
        for (Map.Entry<String, String> entry : v.entrySet())
            System.out.println ("Key = " + entry.getKey()
            + " , value = " + entry.getValue());
    }
}

```



17) for each: -

```
public class {  
    public static void main(String[] args) {  
        Map<String, String> map = new HashMap<String,  
            String>();  
        map.put("Tyo", "pvp");  
        map.put("Mouni", "devarapalli");  
        map.put("Rachana", "Analeepalli");  
        Set<Map.Entry<String, String>> = map.entrySet();  
        map.forEach((k, v) -> System.out.println(  
            "key = " + k + ", value = " + v));  
    }  
}
```

map: -

keyset: Set<String> l = <sup>map</sup>new Keyset();

entryset: - Set<map.entry<String, String>>  
= map.entrySet();

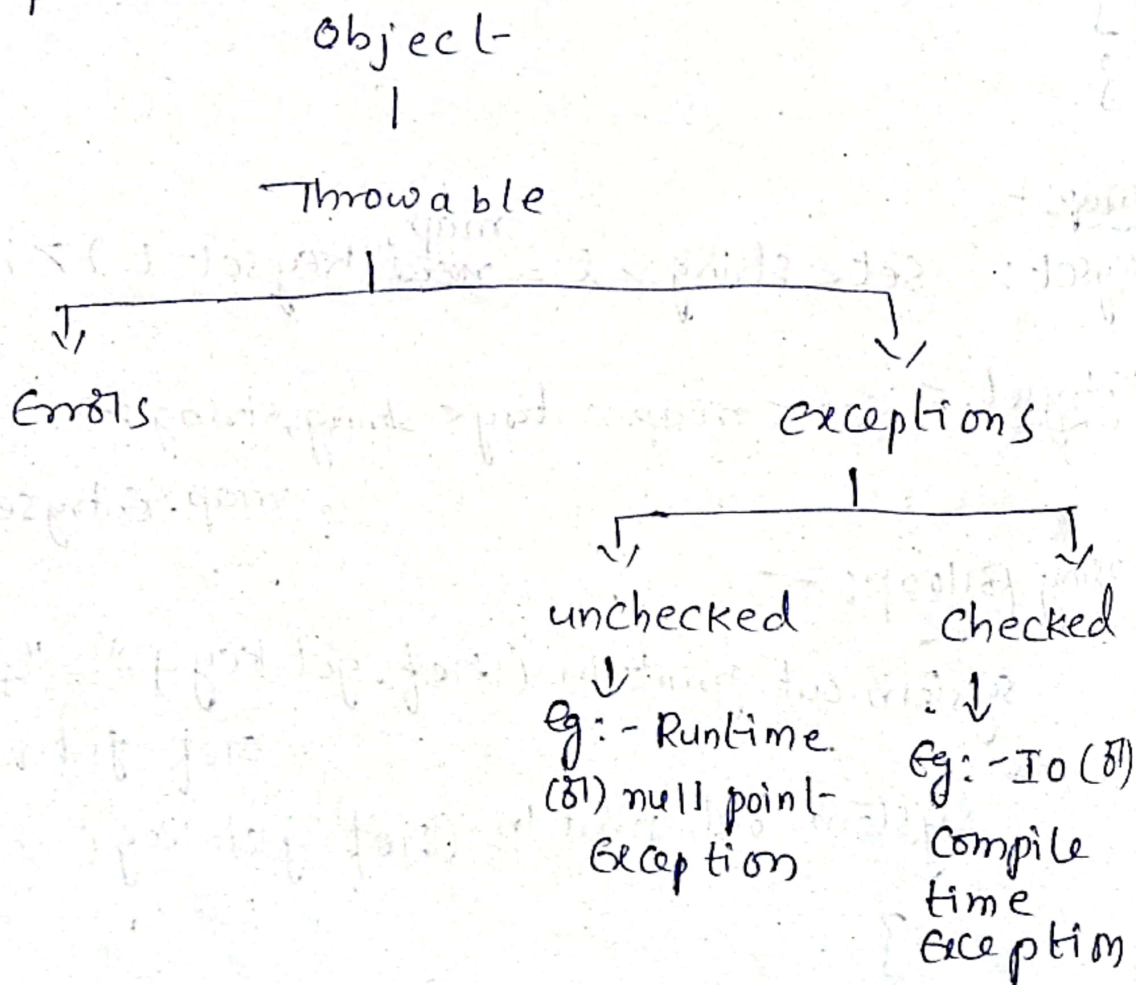
using for loop: -

```
System.out.println(ref.getKey() + " = " +  
    ref.getValue());  
System.out.println(ref.getKey());  
}  
}
```

Iterator for map:-

```
Iterator < map.Entry < string, string > > itr  
= map.entrySet().iterator()  
while (itr.hasNext())  
{  
    set < map.Entry < string, string > > entry = itr.next(),  
    map.Entry < string, string > keyvalue = ref.next()  
    System.out.println (keyvalue.getKey() + " == "  
                        + keyvalue.getValue());  
}
```

Exception:-



Example:-

public class Tjo.<sup>Example</sup> {

public static void main( String[] Args )

{

TjoExample. example = new TjoExample ( );

int i = 0;

try {

i = example.divi(i:10; j=1);

}

catch (Exception.e)

{

e. printStackTrace ( );

}

System.out.println(i+j);

}

public int divi (int i, int j) throws Exception

{

int k = i/j;

return k;

}

}