MOHITH REDDY
192311069
CSA 0985

ASSIGNMENT - 3
JAVA

(4)

21/08/24

Collections of Java as Follows

1. Arraylist :- An Array list is resizable array implement.

```
import Java.util ;
    class Array listen {
  Public static void main (string [ ] args) {
  Araylist <string> list = new array list <> ();
      List . add ("Apple");
      List . add ("Banana");
      List . add ("cherry");
      System . Out . println (List);
  }
}.
```

Output

```
[Apple, Banana, cherry]
```

2. Linkedlist :-

A Linkedlist is a doubly Linked list implementation of list interface.

Program.

```
import Java.util ;
Class Linked list en {
   Public static void main (string args[ ]) {
    Linked list <string> List = new Linkedlist <> ();
        List. add ("Apple");
       List. add ("cherry");
    }
  }
```

Output

```
[Apple, cherry]
```

3. Hashset :-

A Hashset is a implementation that uses a hashtable For storage.

Code :-

```
import Java.util ;
 Class Hashen {
 Public static void main (string args[]){
     Hashset<strings> set = new Hash set<>();
         set.add ("Apple");
         set.add ("Icecream");
         System.out.println(set);
      }
      }
```

Output

```
[Apple, Icecream]
```

4. Tree set

A Tree set is a set implementation that uses a tree For storage.

Code:-

```
import Java.util;
Class Treeseten {
 Public static void main (string args []){
 Treeset <strings> set = new Treeset<>();
       set.add ("Apple");
       set.add ("Banana");
       set.add ("cherry");
     System.out.println (set);

  }
  }
```

Output :-

```
[Apple, Banana, cherry]
```

Hashmap:-

A map implementation that uses a hash table for storage.

```
import Java .until ;
  class hashmap em {
   Public static . Void main (string args[]) {
Hashmap <string ,Integer> map. new hashmap<>();
    map. put ("Apple",1);
    map. put ("Banana", 2);
    map. put ("cherry",3);
    system. out. print-ln (map);
  }
 }
```

Output:-

```
{Apple =1 , Banana= 2, cherry=3}
```

6. Tree Map

A 'Tree Map is a map implementation that uses a tree for storage.

Code :-

```
   import Java util ;
 class Treemapen {
  Public static Void main (string args[]){
 Tree map <string , Integer map =new .Tree map ();
        map . put ("Apple", 1);
        map . put ("Banana", 2);
        map . put ("cherry",3);
       system. out. print ln (map);
    }
   }
```

Output

```
{Apple=1, Banana=2, Cherry=3}.
```

7. Linkedhashset

A Linkedhashset is a set implementation that uses a hashtable and linkedlist for storage.

Code:-

```
import Java until;
  class Linked hashset {
    Public static void main (String args [J){
 Linked hashset <string> set = new linked hash set <> ();
       set.add ("Apple");
       set.add ("Banana");
       set.add ("cherry");
       System.out.printtn (set);
    }
    }
```

Output:-

```
[Apple, Banana, cherry]
```

8. priority Queue:-

A priority Queue is a Queue implementation that orders elements Based on their natural ordering or a Custom Comparator.

Code:-

```
import Java .util;
  class priority Queue {
   Public static void main (string [] args){
 Priority Queue<string> Queue = new priority Queue<>();
      Queue.add("Apple");
      Queue.add ('Banana");
      Queue.add ("cherry");
      system.out.printtn(queue);
   }
  }
```

Out-put

```
[Apple, Banana, Cherry]
```

9. Array Dequeue:-

An Array dequeue is a deque. implementation that uses an array For storage.

Code :-

```
import java. util;
Class Array Deque.{
 Public static Void main (string args []);
 Array Deque < string > deque = new array deque<>();
   deque .add.("Apple");
   deque .add ("Banana");
    system.out.println (dequeue);
    }
   }
```

Output-

```
[Apple, Banana].
```

10. Stack

LIFO implementation of the list interface.

```
import java.util;
  Class stack. {
Public static void main(string args[]){
stack<string> stack = new Stack <>();
  stack.push ("Apple");
  stack.push ("Banana");
  Stack.push ("cherry");
system.out . println (stack);
}
}
```

Output:-

```
[Apple, Banana, cherry].
```

11. Vector

A Vector is a synchronized implementation of the list interface.

Code:-

```
import Java.util ;
 Class Vector {
 Public static void main (string args[]) {
 Vector <string> Vector = new Vector <> ();
       Vector.add ("Apple");
       Vector.add(" Custard Apple ");
  system.out. println (Vector);
  }
 }
```

Output-

```
[Apple, Custard apple]
```