12/8/24 Java programming

Siva chaitanya
192365021
CSA0985

ASSIGNMENT-3

Collections of Java as follows:

1. **Arraylist:** An arraylist is resiazble array implement.

```
import java.util;
class arraylistex{
    public static void main (string[]args){
     Arraylist <string> List = new Arraylist <>();
      List.add ("Banana");
      List.add ("cherry");
  system.out.println(list);
}
}
```

Output:- [Apple, Banana, cherry]

2. **Linkedlist :**

A linked list is a doubly linked list implement of list interface

program:

```
import java.util.*;
class linkedlist ex{
public static void main(string args[]) {
  Linkedlist <string> list =new Linked <str>();
    List.add <"Apple">;
```

```
list.add("cherry");
}
```

Output: (Apple, cherry);

3. Hashset: A hashset is a set implementation that uses a hash table for storage.

Code:

```
import java.util.*;
class hashset{
  public static void main(string[]args)
    hashset<string>set: new hashset<>();
     set.add("Apple");
     set.add("Icecream");
   system.out.println(set);
}
}
```

Output: [Apple, Icecream]

4. Treeset: A treeset is a set implementation that uses a tree for storage.

```
Code: import java.util.*;
 class Treeseter{
  public static void main(string args[]){
  Treeset<string> set =new treeset<>();
    set.add("Apple");
```

```
set.add("Banana");
set.add("cherry");
system.out.println(set);
}
}
```

Output:- (Apple, Banana, cherry)

5. Hashmap:- a map implementation that uses a hash table for storage.

```
import java.util.*;
class hashmapex{
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.println(map);
}
}
```

Output:- {Apple=1, Banana=2, cherry=3}

6. Treemap:-

A tree map is a map implementation that uses a free storage.

Code:- import java.util.*;

```
class Treemapex{
public static void main(string args[]){
```

```
Tree map<string. Integer map = new Treemap <>();
map. put (" Apple ", 1);
map. put (" Banana", 2);
map. put (" cherry", 3);
system. out. println (map);
}
}
```

Output :- {Apple = 1, Banana = 2, cherry = 3}

7. Linked hashset;

A linked hashset is a set implementation that uses a hashtable and linked list for storage.

```
Code :- import java util. *;
class linked hashset ex{
public static void main (string args[]){
Linked hashset <string> set = new linked hashset < >();
set. add ("Apple");
set. add ("Banana");
set. add ("cherry");
system. out. println (set);
}
}
```

Output :-

[Apple, Banana, cherry]

priority Queue: A priority queue is a Queue imple 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.println ("queue");
}
```
Output:- (Apple, Banana, cherry)

9. Array Dequeue:
```
Import java.util.*;
class Array dequeue{
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 list interface

Code :

```
Import Java.util.*;
class stack{
public static void main(string args[]){
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 vectorex{
public static void main(string args[]){
Vector<string> vector = new vector<>();
vector.add("Apple");
Vector.add("mango");
system.out.println(vector);
}
}
```

output:

[Apple, mango]