1. Analyze the code snippet below:

```
package com.spaneos.ct;
import java.util.*;
public class Question {
		public static void main(String[] args) {
			List<String> list=new ArrayList<String>();
			list.add("Manoj");
			list.add("Rajesh");
			Collections.sort(list);
			list.add("Lakshman");
			System.out.println(list);
		}
}
```

What is the output?

A. [Rajesh, Lakshman, Manoj]
B. [Manoj, Rajesh, Lakshman]
C. [Lakshman, Manoj, Rajesh]
D. Compilation Error

2. Analyze the code snippet below and fill in the blanks:

```
package com.spaneos.ct;
import java.util.*;
public class Question {
	public static void main(String[] args) {
	List<String> list=new ArrayList<_____________>();
	list.add("Rajesh");
	list.add("Manoj");
	for(_____________ name:list)
	System.out.println(name);
	}
}
```

A. Object, String
B. Object, Object
C. String, Strings
D. String, String

3. Analyze the code snippet below:

```
package com.spaneos.ct;
import java.util.*;
public class Question {
	public static void main(String[] args) {
```

```
                // Insert Code Here
                int k=1;
                for (int i = 1; i <= 5; i++) {
                        List<Integer> row = new ArrayList<Integer>();
                        for (int j = 1; j <= 5; j++)
                                row.add(k * j);
                        k++;
                        table.add(row);
                }
                for (List<Integer> row : table)
                        System.out.println(row);
        }
}
```

Which statement, if inserted at "// Insert Code Here,  will allow this code to compile and run?

A.  List<Integer> table=new ArrayList<Integer>();
B.  List<List<Object>> table=new ArrayList<List<Integer>>();
C.  List<List<Integer>> table=new ArrayList<List<Integer>>();
D. List<List<Integer>> table=new ArrayList<List<Object>>();

4. Analyze the code snippet below:

```
package com.spaneos.ct;
import java.util.*;
public class Question {
        public static void main(String[] args) {
                Set<Integer> set=new HashSet<Integer>();
                set.add(20);
                set.add(10);
                set.add(20);
                set.add(30);
                int sum=0;
                for(Iterator<Integer> iterator=set.iterator();iterator.hasNext();){
                        sum+=iterator.next();
                }
                System.out.println("The sum is :"+sum);
                }
}What is the output?
```

A. Compliation Error
B. The sum is : 0
C. The sum is : 80
D. The sum is : 60

5. Fill in the blanks

```
package com.spaneos.ct;
import java.util.*;
public class Question {
	public static void main(String[] args) {
		_____________________

		_____________________
	}
	public static void showValues(Set<Object> set){
	}
}
```

A. Set<String> set=new HashSet<String>( );
   showValues(set);
B. Set<Object> set=new HashSet<String>( );
   showValues(set);
C. Set<Object> set=new HashSet<Object>( );
   showValues(set);
D. Set<String> set=new HashSet<Object>( );
   showValues(set);

6. Analyze the code snippet below and fill in the blanks:

```
package com.spaneos.ct;
import java.util.*;
public class Question {
	public static void main(String[] args) {
		Set<String> set=new HashSet<String>();
		set.add("Krish");
		set.add("Rajesh");
		set.add("Manoj");
		_____________<String> iterator=set.iterator();
		__________(iterator._____________){
			String name=iterator._____________;
			System.out.println(name);
		}
	}
}
```

A. ListIterator, for, next( ), hasNext( )
B. Iterator, while, next( ), hasNext( )
C. Iterator, while, hasNext( ), next( )

D. Enumeration, while, hasMoreElements( ),nextElement( )

7. Analyze the code snippet below:

```
package com.spaneos.ct;
import java.util.*;
class Product{
	int pid;
	public Product(int pid) {
		this.pid=pid;
	}
	@Override
	public String toString() {
		return " "+pid;
	}
}
public class Question {
	public static void main(String[] args) {
		Set<Product> set=new TreeSet<Product>();
		set.add(new Product(1003));
		set.add(new Product(1002));
		set.add(new Product(1001));
		System.out.println(set);
	}
}
```

What is the output?

A. Compilation Error
B. [ 1001 ,  1002 , 1003 ]
C. [ 1003,   1002,   1004]
D. Runtime  Exception: java.lang.ClassCastException

8. Analyze the code snippet below:

```
import java.util.*;
public class Question {
	public static void main(String[] args) {
		Set<Integer> set = new HashSet<Integer>();
		Integer i1 = 40;
		Integer i2 = 41;
		set.add(i1);
		set.add(i1);
		set.add(i2); System.out.print(set.size() + " ");
		set.remove(i1); System.out.print(set.size() + " ");
		i2 = 40;
		set.remove(i2); System.out.print(set.size() + " ");
```

```
        }
}
```
What is the output?

A. 2 1 2
B. 2 1 0
C. 2  2 2
D. 2 1 1

9. Analyze the code snippet below:

```
import java.util.*;
public class Question {
        public static void main(String[] args) {
                TreeSet<Integer> s = new TreeSet<Integer>();
                TreeSet<Integer> subs = new TreeSet<Integer>();
                for(int i = 1; i < 10; i++)
                if(i%2 != 0)s.add(i);
                subs = (TreeSet)s.subSet(5, false, 9, true);
                s.add(9);
                System.out.println(s + " " + subs);
        }
}
```
What is the output?

A. [1, 3, 5, 7, 9] [5, 7, 9]
B. [ 3, 5, 7, 9] [5, 7, 9]
C. [1, 3, 5, 7, 9] [ 7, 9]
D. [1, 3, 5, 7] [5, 7, 9]

10. Analyze the code snippet below:

```
import java.util.*;
public class Question {
        public static void main(String[] args) {
                String names[]=new String[]{"Kiran","Mahesh","Rajesh","Lakshman"};
                Arrays.sort(names);
                System.out.print(Arrays.binarySearch(names,"Lakshman"));
                System.out.print(" -  "+Arrays.binarySearch(names, "Mahesh"));
        }
}
```
What is the output?

A. 4 - 2
B. 2 - 4
C. 1 - 2
D. 2 - 1

11. Analyse this code and fill in the blanks to generate the following outupt?

1001 = Rajesh
1002 = Balu

```
public class Question {
	public static void main(String[] args) {
		Map<Integer,String> map=new HashMap<Integer, String>();
		map.put(1001, "Rajesh");
		map.put(1002, "Balu");
		Set<Integer> keys=map._____________;
		for(Integer i:keys)
			System.out.println(i+" = " +map._____________(i));
	}
}
```

A. values( ), put
B. keys( ),get
C. entrySet( ),get
D. keySet( ), get

12.
```
Map<Integer,String> map=new HashMap<Integer, String>();
		map.put(1003, "Rajesh");
		map.put(1001, "Balu");
		map.put(1005, "Balu");
		Set<Integer> keys=map.keySet();
		// Insert Code Here
```

What code, if inserted at "//Insert Code Here", will sort the keys in the keys HashMap?

A. Arrays.sort(keys);
B. keys = new TreeSet(keys);
C. Collections.sort(keys);
D. keys = new SortedSet(keys);

13. What code, if inserted at "//Insert Code Here", will make HashMap synchronized?

```
public class Question {
	public static void main(String[] args) {
		Map<Integer,String> map=new HashMap<Integer, String>();
		map.put(1003, "Rajesh");
```

```
            map.put(1001, "Balu");
            map.put(1005, "Balu");
            // Insert Code Here
        }
}
```

A. Collection.synchronizedMap(map);
B. Collections.synchronizedHashMap(map);
C. Collections.synchronizedMapObject(map);
D. Collections.synchronizedMap(map);

14. Analyze the code snippet below and fill in the blanks:

```
public class Question {
        public static void main(String[] args) {
            Properties properties=System._________________;
            String java_home=(_______________)properties._________("java.home"));
        }
}
```

A. getProperties( ), Object , get
B. getProperties( ), String , get
C. getProperties( ), String, getKey
D. getPropertie( ), Object , get

15. Fill in the blanks :

Collection is a _____, Collections is a _____ and collection is a_____

A. Interface , Class, Interface
B. Interface , Class, Framework
C. Interface , Class, abstract class
D. Interface , Class, root class

16. Which two code fragments, inserted independently at line 2, will compile without warnings?

```
1. public class Example {
2.          //Insert code
3.                  list.add("Spaneos");
4.          }
5. }
```

A. public void addStrings(List list) {
B. public void addStrings(List<String> list) {
C. public void addStrings(List<? super String> list) {
D. public void addStrings(List<? extends String> list) {

17. Which code fragment, inserted at line 2 will generate output as [1, 2, 3]

```
public class Example {

    1.  public static void main(String[] args) {
    2.      //Insert code here
    3.      set.add(new Integer(2));
    4.      set.add(new Integer(1));
    5.      set.add(new Integer(3));
    6.      System.out.println(set);

    }
}
```

A. Set set = new TreeSet();
B. Set set = new HashSet();
C. Set set = new SortedSet();
D. List set = new SortedList();
E. Set set = new LinkedHashSet();

18. What is the output of the following code snippet?

```
public class Example {
    public static void main(String[] args) {
        Map<String,Map<String,String>> map=new HashMap<>();
        Map<String,String> stateInfo=new HashMap<>();
        stateInfo.put("KA", "Bangalore");
        stateInfo.put("AP", "Hyd");
        map.put(" India",stateInfo);
        for(Map.Entry<String, Map<String,String>> entry:map.entrySet()){
            String key=entry.getKey();
            Map<String,String> value=entry.getValue();
            System.out.print(key);
            for(Map.Entry<String, String> entryset:value.entrySet()){
                String k=entryset.getKey();
                if(k.equals("KA"))
                System.out.print(" "+k+" "+entryset.getValue());
            }
        } } }
```

A. India KA Bangalore

AP Hyd

B. India KA Bangalore
   India AP Hyd
C. India AP Hyd
D. India KA Bangalore
E. India Ap Hyd KA Bangalore

19. Which collections classes allow you to grow or shrink its size and provides indexed access to its elements, but whose methods are not synchronized.

A. java.util.HashSet
B. java.util.LinkedHashSet
C. java.util.List
D. java.util.ArrayList
E. java.util.Vector

20. What is the output of the following code snippet?

```
public class Example {
        public static void main(String[] args) {
                Map<Integer, String> player=new HashMap<>();
                player.put(1001, "Dravid");
                player.put(1004, "Sachin");
                player.put(1003, "Dhoni");
                player.put(1002, "Raina");
                Set<Integer> set=player.keySet();
                set=new TreeSet<>(set);
                for(Integer i:set)
                        System.out.print(i+" = "+player.get(i)+" ");
        }
}
```

A. 1001 = Raina 1002 = Sachin  1003 = Dhoni  1004 = Dravid
B. 1004 = Dravid 1003 = Sachin  1002 = Dhoni  1001 = Raina
C. 1001 = Dravid 1002 = Sachin  1003 = Dhoni  1004 = Raina
D. 1001 = Dravid 1002 = Raina  1003 = Dhoni  1004 = Sachin