1. Analyze the code snippet below:

```
package com.core.ct;
import java.util.ArrayList;
import java.util.List;
public class MyClass {
    public static void main(String args[]) {
            List<Object> list=new ArrayList<_________>();
            list.add("Rajesh");
            System.out.println(list);
    }
} Fill in the blank with the appropriate class.
```

A. Object
B. String
C. Number
D. None of the above

2. Analyze the code snippet below:

```
package com.core.ct;
import java.util.*;
public class MyClass</* Insert Code Here*/> {
    private N min, max;
    public N getMin(){
        return min;
    }
    public N getMax(){
        return max;
    }
} What code  inserted at "//Insert Code Here", will ensure that the program compilation is
sucessful
```

A. Integer
B. N extends Integer
C. N extends int
D. ? extends Integer

3. Which of these is a valid annotation declaration

A. @Annotation MyAnnotation
B. @interface MyAnnotation
C. interface   @MyAnnotation
D. @class Myannotation

4. Which of these are valid type of the annotations

A. Single data type, Multiple data type, Double
B. Single-value, Multi-value, Marker
C. Only object types
D. None of the above

5. @override annotation represents

A. Don't override the super class method
B. Sub class should override a method in its super class.
C. Super class should override a method in its sub class.
D. Override the super class method in subclass, by changing the method name in the subclass.

6. Analyze the code snippet below:

```
______ MyExample{

            int value1( ) default ____;
            String value( ) default ____;
} Fill in the blanks
```

A. class
   0
   " "
B. @interface
   10.0
   "Rajesh"
C. @interface
   0
   "Rajesh"
D. @clcass
   0
   " "

7. Which of these is true about the annotations?
   A. Less Code
   B. Better Compile time error detection
   C. Can reduce time on unhandy code-writing and focus more on business logic
   D. All of the above

8. _____________ annotation checks that the method is an override and causes a compilation warning if the method is not found in one of the parent classes.

   A. @Deprecated
   B. @SuppressWarnings
   C. @Override
   D. None of the above

9. Which of these are valid return types for Annotation methods?
   A. int
   B. float
   C. void
   D. A and B but C

10. getDeclaredMethod() returns the following methods

   A. protected methods of the class only
   B. private methods and class method

C. Only inherited methods
D. Only static methods

11. Class.getDeclaredClasses() method returns ?

A. All the extended classes
B. All the extended interfaces
C. All the implemented interfaces
D. A and C

12. Analyse this code snippet and fill in the blanks

```
    public class SystemInfo {
            public static void main(String[] args) {
            try{
            ___________ c=Class.___________("java.lang.ArithmeticException");
            ArithmeticException s=(_____________)c.newInstance();
            System.out.println(s.toString());
            }catch(Exception e){
                        e.printStackTrace();
            }
            }
    }
```

A. Class, forname, ArithmeticException
B. class, forname, ArithmeticException
C. Class, forName, ArithmeticException
D. None of the above

13. getDeclaredFields() returns

A. public, protected, default (package) access, and private fields but excludes inherited fields.
B. public, protected, default (package) access, and private fields and includes inherited fields.
C. private fields but excludes inherited fields.
D. None of the above

14. getDeclaredConstructors() Method returns

A. An array of Constructor objects reflecting all the constructors declared by the class represented by this Class object.
B. An array of Constructor objects reflecting all the constructors declared by the class and super class represented by this Class object.
C. An array of Constructor objects reflecting all the public constructors declared by the class, represented by this Class object.
D. None of the above

15. Which of these are valid Generic Declaration in java 1.7?

A. List<Object> list=new ArrayList<>;
B. Listlist=new ArrayList<Object>();
C. List< > list=new ArrayList<String>();
D. List<Object> list=new ArrayList< >();