```
++++++++++++++++++++++++++
Overriding w.r.t variables
++++++++++++++++++++++++++
 => Overriding of variables is not possible in java
 => In case of variables Overriding, compiler will bind the value of the variable
based on the reference type.

eg#1.
class Parent{
       int x = 888;
}
class Child extends Parent{
      static int x = 999;
}

Parent p = new Parent();
System.out.println(p.x);//888

Child c1 = new Child();
System.out.println(c1.x);//999

Parent p1 = new Child();
System.out.println(p1.x);//888

What is the difference b/w
      ArrayList al =new ArrayList();
      [Child c =new Child();]
       If we know runtime object type exactly, then we need to use this approach.
       By using child reference, we can make a call to "parent class and child
class methods".

            vs
      List l = new ArrayList();
      [Parent p =new Child();]
       If we don't know exactly the runtime object, then we need to use approach.
       By using parent reference, we can make a call to "parent class methods
only".
      Note: second apporach is commonly used in realtime coding.

Note::
abstract class Parent
{
      public abstract void m1();
}
class Child extends Parent
{
      @Override
      public void m1(){

      }

      public void m2(){

      }
}

Parent p = new Child();
      p.m1();
      p.m2();//CE
```

```
Eg#1.
class Parent
{
      public void m1(int... i){
            System.out.println("FROM PARENT");
      }
}
class Child extends parent
{
      public void m1(int i){//Not overriding,it is overloading[Compiler]
            System.out.println("FROM CHILD");
      }
}

Parent p = new Parent();
 p.m1(10);//FROM PARENT

Child c =new Child();
 c.m1(10);//FROM CHILD

Parent p1 =new Child();
  p1.m1(10);//FROM PARENT

Eg#2.
class Parent
{
      public void m1(int... i){
            System.out.println("FROM PARENT");
      }
}
class Child extends parent
{
      public void m1(int... i){//Overriding[JVM -> RuntimeObject]
            System.out.println("FROM CHILD");
      }
}

Parent p = new Parent();
 p.m1(10);//FROM PARENT

Child c =new Child();
 c.m1(10);//FROM CHILD

Parent p1 =new Child();
  p1.m1(10);//FROM CHILD

In how many ways we can create an object for java class?
 a. using new operator
      Test t  =new Test();
 b. using reflection api
      Test t = (Test)Class.forName("Test").newInstance();
 c. using clone apporach
      Test t1 =new Test();
      Test t2=(Test)t1.clone();
 d. using factory methods
      Runtime r   = Runtime.getRuntime();
      DateFormat f= DateFormat.getInstance();
```

e. using DeSerialization
Test t1=(Test)new ObjectInputStream(new FileInputStream("abc.ser")).readObject();

Q>
Given:

```
1. public class Barn {
2.    public static void main(String[] args) {
3.          new Barn().go("hi", 1);
4.          new Barn().go("hi", "world", 2);
5.    }
6.    public void go(String... y, int x) {
7.          System.out.print(y[y.length - 1] + " ");
8.    }
9. }
```

What is the result?
A. hi hi
B. hi world
C. world world
D. Compilation fails.//Answer
E. An exception is thrown at runtime.

QUESTION
What is the result?

```
11. public class Person {
12.   String name = "No name";
13.   public Person(String nm) { name = nm; } //super()
14. }
15.
16. public class Employee extends Person {
17.   String empID = "0000";
18.   public Employee(String id) { empID = id; }//super()
19. }
20.
21. public class EmployeeTest {
22.   public static void main(String[] args){
23.         Employee e = new Employee("4321");
24.         System.out.println(e.empID);
25.   }
26. }
```

Choose the answer
A. 4321
B. 0000
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 18. //Answer

Given:

```
1. class Atom {
2.    Atom() { System.out.print("atom "); }//super()// 1. atom
3. }
4. class Rock extends Atom {
5.    Rock(String type) { System.out.print(type); }//super() //-----2. granite
6. }
7. public class Mountain extends Rock {
8.          Mountain() {
```

```
9.                super("granite ");
10.               new Rock("granite ");
11.         }
12.   public static void main(String[] a) { new Mountain(); }
13.}
```

What is the result?
A. Compilation fails.
B. atom granite
C. granite granite
D. atom granite granite
E. An exception is thrown at runtime.
F. atom granite atom granite

Answer: F

Given:

```
interface TestA { String toString(); }
      public class Test {
                  public static void main(String[] args) {
                              System.out.println(new TestA() {
                                                  public String toString() {
                                                        return "test";
                                                   }
                                              }
                               );
                  }
}
```

What is the result?
A. test//Answer
B. null
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 1.
E. Compilation fails because of an error in line 4.
F. Compilation fails because of an error in line 5.

Given:

```
11. abstract class Vehicle { public int speed() { return 0; }
12. class Car extends Vehicle { public int speed() { return 60; }
13. class RaceCar extends Car { public int speed() { return 150; } ...
21.   RaceCar racer = new RaceCar();
22.   Car car = new RaceCar();
23    Vehicle vehicle = new RaceCar();
24    System.out.println(racer.speed() + ", " + car.speed() + ", " +
vehicle.speed());
```

What is the result?
A. 0, 0, 0
B. 150, 60, 0
C. Compilation fails.
D. 150, 150, 150
E. An exception is thrown at runtime.

Answer: D

Given:

```
21. class Money {
22.         private String country = "Canada";
23.         public String getC() { return country; }
24. }
```

```
25. class Yen extends Money {
26.         public String getC() { return super.country; }
27. }
28. public class Euro extends Money {
29.         public String getC(int x) { return super.getC(); }
30.         public static void main(String[] args) {
31.                     System.out.print(new Yen().getC() + " " + new
Euro().getC());
32.               }
33. }
```

What is the result?

A. Canada
B. null Canada
C. Canada null
D. Canada Canada
E. Compilation fails due to an error on line 26.//Answer
F. Compilation fails due to an error on line 29.
G. Compilation fails due to an error on line 32.

Explain this keyword and super keyword in java?

this keyword -> To avoid name clash b/w local and instance variables of a class inside a method.

It holds the address the current object, which is active on the heap.

super keyword => To avoid name clash b/w parent class instance variable and child class instance variable.

it always refers to the parent class members.

Given:

```
1. public class Boxer1{
2.          Integer i; // i =null
3.          int x;     // x = 0
4.          public Boxer1(int y) {
5.                x = i+y;// x = null + 4
6.                System.out.println(x);
7.          }
8.          public static void main(String[] args) {
9.                new Boxer1(new Integer(4));
10.         }
11.}
```

What is the result?

A. The value "4" is printed at the command line.
B. Compilation fails because of an error in line 5.
C. Compilation fails because of an error in line 9.
D. A NullPointerException occurs at runtime.//Answer
E. A NumberFormatException occurs at runtime.
F. An IllegalStateException occurs at runtime.

Given:

```
10. public class SuperCalc {
11.         protected static int multiply(int a, int b) { return a * b;}
12. }
```

and:

```
20. public class SubCalc extends SuperCalc{
21.         public static int multiply(int a, int b) {
22.               int c = super.multiply(a, b);//super ---> object, static ->
classdata
```

```
23.               return c;
24.         }
25. }
```
and:
```
30. SubCalc sc = new SubCalc ();
31. System.out.println(sc.multiply(3,4));
32. System.out.println(SubCalc.multiply(2,2));
```

What is the result?
A. 12
B. The code runs with no output.
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 21.
E. Compilation fails because of an error in line 22.//Answer
F. Compilation fails because of an error in line 31.