Final:

R1: We can make final Primitive variable and final Reference variable.

R2: Constructor can not be a final.

R3: If we make any class final that means it can not be Inherited. To stop inheritance we make a class final.

R4: If we make any method of a class as final then it can not be overridden in child class.

R5: If we make any local variable as a final then it will become constant that means we can not change the value of the variable through out the function.

R6: We can also make formal argument as final.

R7: If we make any static data member of a class as a final then it will become constant that means we can not change the value of the varaible through out the program and it has to be initialized at class level.

R8: If we make any non-static data member of a class as a final then it will become constant that means we can not change the value of this variable through out the program and it as to be initialized at class level.

R9: If we want to make any non-static data member as a blank final variable then it has to be initialized via constructor.

Program:

```
// Final Class, Final Member Function, Final Local Variable, Final Data Member

final class Final{ // it can not be inherited

    // Final Class Variable
    // (Blank Final Variable) can only assign here and inside constructor
    // It does not have any value like other class level variable

    final int x = 100; // can't change this
    final int y;       // can assign only once in constructor


    int n; // it has by defualt value 0

    final static int m = 900; // can't change it through out the program.
    final static int p;      // it static and has by default 0 so cant change it.
                                 It has to initialized/load at class level.
```

```java
`   // Constructor can't be final
    Final(){

        //initilizing final y.
        y = 200;
        System.out.println(x + " And "+ y);
    }

    void show(){

        // Final Local Variable
        final int z = 300;
        //z = 400; // Can't re-assign
        System.out.println("Final Parent Show. Z = " + z);
    }

    void display(final int a){    // can't change the argument value
        //a = 600;  // Can't Change it
        System.out.println("A is : " + a);
    }

    public static void main(String...a){
        final Final f = new Final(); //Now we can't change it
        //f = new Final(); // will create error.
        f.show();
        f.display(500);
    }

}

//This will give an compile time error

/*class Child extends FinalParent{

}*/
```

Program: Blank Final

```java
class BlankFinal{

    final int x;   // Blank Final Variable, no value inside
    final int y;   // Blank Final Variable, no value inside
                   // It has to initialize only inside constructor.

    int z;     // non-final, by default value 0

    BlankFinal(int x){
        this.x = x;
        y = 500;
        System.out.println("X is: " + x);
        System.out.println("Y is: " + y);
        System.out.println("Z is: " + z);
    }

    public static void main(String...a){
        BlankFinal bf = new BlankFinal(200);
        System.out.println("Inisde Main X is: " + bf.x);
        // bf.x = 600; // Can't change it. error

        BlankFinal bf2 = new BlankFinal(400);
        System.out.println("Inside Main with BF2 X is: " + bf2.x);
    }
}
```

Program: Final with Method

```
class FinlaWithFunction{

    //Non-static Intialization
    final int x = getX();     // non-static function
    final int y = getY();     // static function works

    //Static initialization
    // final static int m = getM(); // non-static function // Not Working demands
        static function

    final static int n = getN(); // Static function

    int getX(){
        return 300;
    }

    static int getY(){
        return 500;
    }

    int getM(){
        return 700;
    }

    static int getN(){
        return 800;
    }

    public static void main(String...a){
        FinlaWithFunction fwf = new FinlaWithFunction();
        System.out.println("From Main Value of X is: " + fwf.x);
        System.out.println("Value of Y is: " + fwf.y);

        System.out.println("Value of N is: " + fwf.n);
    }
}
```

Ajab Java Ki Gajab Kahani:

If we initialize final class data member (static/Instance) through function then it first assigns default value and then assign the value that function returns. But Java rule says we can only assign once to final variables. Refer the program for more details.

```
class FinalWithTwist{

    final int x = getValue();
    final static int y = getY();

    int getValue(){

        //It should create error but it prints 0 then return 500
        System.out.println("From getValue X is: " + x);
        return 500;
    }

    static int getY(){
        System.out.println("From getValue Y is: " + y);
        return 900;
    }

    FinalWithTwist(){
        // x is assigned two times first 0 and then 500.
        //but final rule says it can only assign one time.

        System.out.println("From Constructor X is :" + x);
        System.out.println("From Constructor y is :" + y);
    }

    public static void main(String...a){
        new FinalWithTwist();
    }

}

/*
Output:
From getValue X is: 0
From Constructor X is :500

Just accepts that it is working. No Answer by Manish Sir
*/
```