void keyword in java:

1. A void method cannot return value

2. In void method we an use only return keyword

3. return keyword is optional, If we donot use that inside a method the program would still execute

4. If a void method is called from System.out.println() then we would get an error, because

System.out.println() cannot print control, it can print only vlues

not a void method

1. These methods return values

2. These methods can be called from System.out.println();

3. If we write any code after "return value" statement then we will get unreachable code error

Example 1:

```
package methodsexamples;

public class A {
        public static void main(String[] args) {//Rule 1 Starts
                A a1 = new A();
                int i = a1.test();
                System.out.println(i);
        }

        public int test(){
                return 100;
        }
}
```

Output: 100

Example 2:

```
package methodsexamples;

public class A {
	public static void main(String[] args) {//Rule 1 Starts
		A a1 = new A();
		float i = a1.test();
		System.out.println(i);
	}

	public float test(){
		return 10.3F;
	}
}
```

Output: 10.3

Example 3:

```
package methodsexamples;

public class A {
	public static void main(String[] args) {//Rule 1 Starts
```

```
		A a1 = new A();

		System.out.println(a1.test());

	}


	public float test(){

		return 10.3F;

	}

}
```

Output:

```
10.3
```

Example 4:

```
package methodsexamples;


public class A {

	public static void main(String[] args) {//Rule 1 Starts

		A a1 = new A();

		System.out.println(a1.test());

	}


	public float test(){

		System.out.println("From test()");

		return 10.3F;

	}

}
```

Output:

```
From test()

10.3
```

Example 5:

```
package methodsexamples;


public class A {

        public static void main(String[] args) {//Rule 1 Starts

                A a1 = new A();

                System.out.println(a1.test());

        }


        public float test(){

                return 10.3F;

                System.out.println("From test()");//Error

        }

}
```

Output

We will get an error unreachable code, because in the above example "System.out.println("From test

()");" will never execute

Example 6:

```
package methodsexamples;
```

```
public class A {
        public static void main(String[] args) {//Rule 1 Starts
                A a1 = new A();
                a1.test();
        }

        public float test(){

        }
}
```

Output:

We will get an error because if a method is not a void then adding return keyword with value is must.

Example 7:

```
package methodsexamples;

public class A {
        public static void main(String[] args) {//Rule 1 Starts
                A a1 = new A();
                a1.test();
        }

        public void test(){

                return 100;
        }
}
```

Output:

Error because void methods cannot return any value

Example 8:

```
package methodsexamples;

public class A {
	public static void main(String[] args) {//Rule 1 Starts
		A a1 = new A();
		a1.test();
	}

	public void test(){
		return;
	}
}
```

Output:

Nothing but will compile and run successfully

Example 9:

```
package methodsexamples;

public class A {
        public static void main(String[] args) {//Rule 1 Starts

                A a1 = new A();

                System.out.println(a1.test()); //Cannot print control

        }


        public void test(){

                return;

        }

}
```

Output:

Error

Example 10:

```
package methodsexamples;


public class A {

        public static void main(String[] args) {//Rule 1 Starts

                A a1 = new A();

                a1.test();

        }


        public void test(){

                return;

                System.out.println("From Test");//Error

        }
}
```

Output:

Unrachable code error

Example 11:

```
public class A {

    public static void main(String[] args) {//Rule 1 Starts

        A a1 = new A();

        long j = a1.test();

        System.out.println(j);

    }


    public int test(){

        return 100;

    }

}
```

Output: 100

Note We are doing here upcasting by copying the data from int to long

Example 12:

```
public class A {

    public static void main(String[] args) {//Rule 1 Starts

        A a1 = new A();
```

```
		byte j = a1.test();//Error
		System.out.println(j);
	}

	public int test(){
		return 100;
	}
}
```

Output:

Error, because we are copying data from bigger memory int to smaller memory byte, and this process is

not automated

Example 13:

```
public class A {
	public static void main(String[] args) {//Rule 1 Starts
		A a1 = new A();
		byte j = (byte)a1.test();
		System.out.println(j);
	}

	public int test(){
		return 100;
	}
}
```

Output:

```
100
```