What gets printed when the following program is run?

```java
public class While {
    public static void main( String[] args ) {
        final int MAX = 8;
        int i = 2, j = -2;
        System.out.println( i + " " + j );
        while ( i <= MAX ) {
            j = i + 1;
            while ( j < MAX ) {
                j += 3;
                System.out.println( i + " " + j );
            }
            i += 2;
        }
        System.out.println( i + " " + j );
    }
}
```

Trace the following program and specify its output.

```java
public class Trace {
    public static void main( String[] args ) {
        System.out.println( "main1" );
        foo1();
        System.out.println( "main2" );
        foo3();
        System.out.println( "main3" );
        foo2();
    }

    public static void foo1() {
        foo3();
        System.out.println( "A" );
    }

    public static void foo2() {
        System.out.println( "B" );
    }

    public static void foo3() {
        System.out.println( "C" );
        foo2();
    }
}
```

We usually define instance variables with the access modifier ______________ while we usually define ctors and methods we want to be part of this type's published interface with the access modifier ______________.

JButton is an example of a GUI ______________ while a JPanel is an example of a GUI ______________ which has a layout manager.
Given the following definitions:

```java
public interface Printable {
    public static final String ORIENTATION = "Portrait";
    public abstract void print( boolean doubleSided );
}
```

class Thing1 implements Printable {
    public Thing1() {
        // ctor initialization here
    }
    public void print( boolean doubleSided ) {
        // print either single/double sided
    }
    public void print() {
        // print single sided by default
        this.print( false );
    }
}

class Thing2 implements Printable {
    public Thing2() {
        // ctor initialization here
    }
    public void print( boolean doubleSided ) {
        // print either single/double sided
    }
    public void print( String orientation ) {
        ORIENTATION = orientation; /* A */
        // print single sided by default
        this.print( false );
    }
}
```

And the following variable definitions and code are in some other class:

```java
Thing1 thing1;
Thing2 thing2;
Printable printable;
```

Indicate which are valid Java statements. Consider each statement executed sequentially in the order it appears.

1) Valid Java statement – No Compiler Error
2) Invalid Java statement – Compiler Error

```java
ing = new Thing1(); _______
g1.print(); _______
g1.print( true ); _______
g1.print( "Landscape" ); _______
ting2 = new Thing2(); _______
ting2.print(); _______
ting2.print( true ); _______
ting2.print( "Landscape" ); _______
```

```java
printable = new Thing1(); _______
printable.print(); _______
printable.print( true ); _______
printable = thing2; _______
printable.print( "Landscape" ); _______
printable.print( true ); _______
printable = new Printable(); _______
thing1 = thing2; _______
thing1 = printable; _______
```

Hint: What does the compiler know about any reference variable at compile time (vs. runtime)?