```
class Color {
     int red, green, blue;

     Color() {
             Color(10, 10, 10);
     }

     Color(int r, int g, int b) {
             red = r;
             green = g;
             blue = b;
     }

     void printColor() {
             System.out.println("red: " + red + " green: " + green + " blue: " +
blue);
     }

     public static void main(String [] args) {
             Color color = new Color();
             color.printColor();
     }
}
```

A. Compiler error: cannot find symbol.

B. Compiles without errors, and when run, it prints the following: red: 0 green: 0 blue: 0.

C. Compiles without errors, and when run, it prints the following: red: 10 green: 10 blue: 10.

D. Compiles without errors, and when run, crashes by throwing `NullPointerException`.

In the FunPaint application, you need to code classes to draw rectangles. A rectangle can have plain or rounded edges. You also need to color a (plain or rounded) rectangle. How will you define classes for creating these plain, colored, and rounded rectangles? You can use is-a relationships as needed.

Look at the following option to implement the required functionality:

```
class Rectangle { /* */ }
class ColoredRectangle extends Rectangle { /* */ }
class RoundedRectangle extends Rectangle { /* */ }
class ColoredRoundedRectangle extends ColoredRectangle, RoundedRectangle { /* */ }
```

A. Compiler error: '{' expected cannot extend two classes.

B. Compiles without errors, and when run, crashes with the exception `MultipleClassInheritanceException`.

C. Compiles without errors, and when run, crashes with the exception `NullPointerException`.

D. Compiles without errors, and when run, crashes with the exception `MultipleInheritanceError`.

In the FunPaint application, you can fill colors to various shape objects. To implement it, you need to implement a `Color` class. The `Color` class has three members, `m_red`, `m_green`, and `m_blue`. Focus on the `toString()` method and check if it works fine.

Choose the best option based on the following program:

```
class Color {
     int red, green, blue;

     Color() {
             this(10, 10, 10);
     }


         Color(int r, int g, int b) {
                red = r;
                green = g;
                blue = b;
         }
         public String toString() {
              return "The color is: " + red + green + blue;
         }

         public static void main(String [] args) {
                 // implicitly invoke toString method
                 System.out.println(new Color());
         }
    }
```

A. Compiler error: incompatible types.

B. Compiles without errors, and when run, it prints the following: The color is: 30.

C. Compiles without errors, and when run, it prints the following: The color is: 101010.

D. Compiles without errors, and when run, it prints the following: The color is: red green blue.

Choose the best option based on the following program:

```
class Color {
     int red, green, blue;

     Color() {
             this(10, 10, 10);
     }

     Color(int r, int g, int b) {
             red = r;
             green = g;
             blue = b;
     }

     String toString() {
             return "The color is: " + " red = " + red + " green = " + green +
" blue = " + blue;
     }

     public static void main(String [] args) {
             // implicitly invoke toString method
             System.out.println(new Color());
     }
}
```

A. Compiler error: attempting to assign weaker access privileges; `toString` was public in `Object`.

B. Compiles without errors, and when run, it prints the following: The color is: red = 10 green = 10 blue = 10.

C. Compiles without errors, and when run, it prints the following: The color is: red = 0 green = 0 blue = 0.

D. Compiles without errors, and when run, it throws `ClassCastException`.