Some Thought Problems in Java

What is the output of the following Java code when `main()` is run?

```
class Changers {
    public static void changeMe1(int value) {
        value = 2*value;
    }
    public static void changeMe2(int[] value) {
        value = new int[value.length];
    }
    public static void changeMe3(int[] value) {
        value[0] = 2*value[0];
    }
    public static void main(String[] args) {
      int x = 23;
      int[] y = { 42 };    // what is this??
      int[] z = { 37 };
      changeMe1(x);
      System.out.println(x);    // Output: __23__ #1 – can't change x directly
      changeMe2(y);
      System.out.println(y[0]);    // Output: __42___ #2 – can't change y directly
      changeMe3(z);
      System.out.println(z[0]); // Output: __74___ #3 – can change any element in z
    }
}
```

Some Thought Problems in Java

What is the output of the following Java code fragment? Briefly justify your answer.
(Recall that `x instanceof C` tells you whether the dynamic type of `x` is either `C` or a subtype of `C`. Also recall that `boolean` values print as `true` or `false`.)

```
class Coin {
   private double value;
   public void setValue(double value) { this.value = value; }
   public double getValue() { return value; }
   public Coin(double value) { setValue(value); }
   public boolean equals(Object that) {
       return (this == that) || (that instanceof Coin) &&
               (this.getValue() == ((Coin) that).getValue());
   }
}
//...
   Coin nickel1 = new Coin(0.05);
   Coin nickel2 = nickel1;
   System.out.println(nickel1.equals(nickel2) + " " + (nickel1 == nickel2));
   // prints true true because nickel1 and nickel2 are the same object
   nickel2.setValue(0.10);
   System.out.println(nickel1.equals(nickel2) + " " + (nickel1 == nickel2));
   // prints true true because nickel1 and nickel2 are the same object
```

Some Thought Problems in Java

What is the output of the following Java code fragment, assuming the same definition of class `Coin` from the previous subproblem? Briefly justify your answer.

```
class Coin {
    private double value;
    public void setValue(double value) { this.value = value; }
    public double getValue() { return value; }
    public Coin(double value) { setValue(value); }
    public boolean equals(Object that) {
        return (this == that) || (that instanceof Coin) &&
                (this.getValue() == ((Coin) that).getValue());
    }
}
//...
   Coin nickel1 = new Coin(0.05);
   Coin nickel2 = new Coin(0.05);
   System.out.println(nickel1.equals(nickel2) + " " + (nickel1 == nickel2));
   // prints true false because both coins have the same value, but are
     distinct objects
   nickel2.setValue(0.10);
   System.out.println(nickel1.equals(nickel2) + " " + (nickel1 == nickel2));
   // prints false false because the coins have different values, and are
     distinct objects
```