AP Homework

Here are various unsavory sundries I have spotted in your code. Please tell me about them.

(1) What is this student's confusion about how the arguments in a method are supposed to work? [Note: if you are this student, talk to someone!]

```
public static boolean isDivisible(int n, int m) {
      String response;
      response = JOptionPane.showInputDialog("Enter a value for n");
      n = Integer.parseInt( response );

      response = JOptionPane.showInputDialog("Enter a value for m");
      m = Integer.parseInt( response );

      if (n % m == 0)
            return true;

      return false;
}
```

(2) Why is this not DRY? Make it dry. (You can write exactly 3 lines of code equivalent to what's below).

```
if (playerTurn == 0){
      System.out.println("It's player 1's turn.");
      response =  JOptionPane.showInputDialog("Which pile would you like to draw from?");
      doTurn(response, pile1, pile2, pile3, pile4, pileTotal);
}

if (playerTurn == 1) {
      System.out.println("It's player 2's turn.");
      response =  JOptionPane.showInputDialog("Which pile would you like to draw from?");
      doTurn(response, pile1, pile2, pile3, pile4, pileTotal);
}
```

(3) Why is this not DRY? Make it dry. (You can write exactly 3 lines of code equivalent to what's below).

```
if (userResponse.equals("pile 1")) {
      userNumber = getNumber();
      pile1 = pileNumber(pile1, userNumber, "pile 1");
      pileTotal = pTotal(pileTotal, userNumber);
} else if (userResponse.equals("pile 2")) {
      userNumber = getNumber();
      pile1 = pileNumber(pile1, userNumber, "pile 2");
      pileTotal = pTotal(pileTotal, userNumber);
} else if (userResponse.equals("pile 3")) {
      userNumber = getNumber();
      pile1 = pileNumber(pile1, userNumber, "pile 3");
      pileTotal = pTotal(pileTotal, userNumber);
} else if (userResponse.equals("pile 4")) {
      userNumber = getNumber();
      pile1 = pileNumber(pile1, userNumber, "pile 4");
      pileTotal = pTotal(pileTotal, userNumber);
}
```

(4) Why is System.out.println a less useful thing to do in this method? What would be a better thing to do? Why?

```
public static void distance(double x1, double y1, double x2, double y2) {
      double deltaX = x2 - x1;
      double deltaY = y2 - y1;
      double dist = sumSquares(deltaX, deltaY);
      double distance = Math.sqrt(dist);
      System.out.println(distance);
}
```

(5) Write a method countDigits(int n) which will return the number of digits in *n*. For example:

```
countDigits(523);      // returns 3
countDigits(9837);     // returns 4
```

(6) Write a method containsEvenDigit(int n) which returns true if any of the digits in *n* are even. For example:

```
containsEvenDigit(3975);          // returns false
containsEvenDigit(2975);          // returns true
```

(7) Write a method containsAllEvenDigits(int n) which returns true if all of the digits of *n* are even. For example:

```
containsAllEvenDigits(264);       // returns true
containsAllEvenDigits(27666);     // returns false
```

(8) In class we started a brute-force solution to find the integer x such that x^2 is a number of the form 9_8_7_6_5_4_3_2_1_0 where each _ is a single digit.

Write a method isSolution(long n) which returns true if its input has that form, false otherwise.

In other words:

```
isSolution(9585756555453525150);    // returns true
isSolution(27);                     // returns false
```

Note #1: If you want to create a long number you can say 29348098234**L** to force it to be long.
Note #2: Think about how to use / and % cleverly so you don't actually need note #1.