**Lab “Methods with Parameters”**

**You should have ONE program. Your program should contain the following methods and the main program below should run them. They MUST be in separate methods. You may call the methods with hard-coded numbers. You do NOT have to use a Scanner.**

**Remember:**

* Methods may be written to accept parameters, which are values given from the calling code into the method. Parameters allow data values to flow into the method, which can change the way the method executes. A method declared with a set of parameters can perform an entire family of similar tasks instead of exactly one task.
* When primitive values such as type int or double are passed as parameters, their values are copied into the method. Primitive parameters send values into a method, but not out of it; the method can use the data values but cannot affect the value of any variables outside of it.
* Two methods can have the same name if they declare different parameters. This is called overloading.
* Methods can be written to return a value to the calling code. This allows methods to perform a complex computation and then provide its result back to the calling code. The type of the return value must be declared in the method's header and is called the method's return type.

1. Write a static void method named lattice that accepts an int number as a parameter and prints that many rows of a lattice. The following outputs are the results of the two calls to the method:

lattice(1) and lattice(3);

**\* \* \* \*  
 \* \* \* \* \* \* \* \*  
\* \*\* \*\* \*\* \*  
 \* \* \* \* \* \* \* \*  
 \* \* \* \***

**\* \* \* \*  
 \* \* \* \* \* \* \* \*  
\* \*\* \*\* \*\* \*  
 \* \* \* \* \* \* \* \*  
 \* \* \* \*  
 \* \* \* \*  
 \* \* \* \* \* \* \* \*  
\* \*\* \*\* \*\* \*  
 \* \* \* \* \* \* \* \*  
 \* \* \* \*  
 \* \* \* \*  
 \* \* \* \* \* \* \* \*  
\* \*\* \*\* \*\* \*  
 \* \* \* \* \* \* \* \*  
 \* \* \* \***

1. Write a set of static double methods that provide conversion between various units of measurement.

|  |  |
| --- | --- |
| static double | **[inchesToFeet](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(double))**(double inches)  Returns the conversion from inches to feet. |
| static double | **[feetToInches](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(float))**(double feet)  Returns the conversion from feet to inches. |
| static double | **[feetToYards](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(double))**(double feet)  Returns the conversion from feet to yards. |
| static double | **[yardsToFeet](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(float))**(double yards)  Returns the conversion from yards to feet. |
| static double | **[feetToMiles](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(double))**(double feet)  Returns the conversion from feet to miles. |
| static double | **[milesToFeet](https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html" \l "abs(float))**(double miles)  Returns the conversion from miles to feet. |

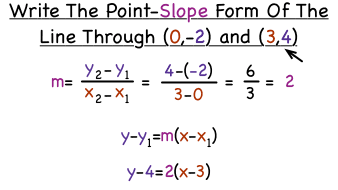
The following output can be used to test your methods:

**114 inches in feet is 9.5  
 12 feet in inches is 144.0  
 32 feet in yards is 10.666666666666666  
 9 yards in feet is 27.0  
 12000 feet in miles is 2.272727272727273  
 7.5 miles in feet is 39600.0**

1. Write three static String overloaded methods:

* getLineEquation that takes two double parameters that represent the slope and y-intercept. Returns a String: y = m x + b example: y = 5.2 x + 3.7
* getLineEquation – takes three double parameters that represent the slope and a point (x1, y1). Returns a String: y – y1 = m(x – x1) example: y – 4.2 = 3.8(x – 6.1)
* getLineEquation – takes four double parameters that represent 2 points (x1, y1) and ( x2, y2). Returns a String: y – y1 = m(x – x1) example: y – 4.2 = 3.8(x – 6.1) To recall how to calculate the slope, see equation below.

Example of point-slope form:



The following output can be used to test your methods:

**y = 5.2x + 3.7**

**y - 2.5 = 4.5(x - 3.1)  
y - 4.0 = 2.0(x - 3.0)**

You can use the main method below to test your work.

public static void main(String[] args) {

lattice (1);  
 System.out.println("\n");  
 lattice (5);  
 int inches = 114;  
 int feet1 = 12;  
 int feet2 = 32;  
 int yards = 9;  
 int feet3 = 12000;  
 double miles = 7.5;  
 System.out.println(inches + " inches in feet is " + inchesToFeet(inches));  
 System.out.println(feet1 + " feet in inches is " + feetToInches(feet1));  
 System.out.println(feet2 + " feet in yards is " + feetToYards(feet2));  
 System.out.println(yards + " yards in feet is " + yardsToFeet(yards));  
 System.out.println(feet3 + " feet in miles is " + feetToMiles(feet3));  
 System.out.println(miles + " miles in feet is " + milesToFeet(miles));  
 System.out.println("\n");  
 String eq1=getLineEquation(5.2, 3.7);  
 System.out.println(eq1);  
 String eq2=getLineEquation(4.5, 3.1, 2.5);  
 System.out.println(eq2);  
 String eq3=getLineEquation(0, -2, 3, 4);  
 System.out.println(eq3);

}