Project 2: Conversion program

Contents

New concepts: Exception handling, arrays, methods (functions), basic class implementation.

1 Program description

Write a program to convert between feet and inches and centimeters. Use the conversion factor 1in = 2.54cm. After the conversion, the menu will be presented again.

2 Menu options

- `0 - Quit`
  Quit the program.
- `1 - Convert feet and inches to centimeters`
  The user enters the number of feet and the number of inches, which are then converted into centimeters.
- `2 - Convert centimeters to feet and inches`
  The user enters the number of centimeters, which is then converted into feet and inches. The number of feet must be integer.
- `3 - Print previous conversions`
  Prints the previous output from commands 1 and 2 in the order they were entered.

3 Error messages

- `ERROR: Input must be an integer in [<LB>, <UB>]!`
  where `LB` and `UB` are lower and upper bounds. If UB is equal to the maximum storable value of an `int`, then "`infinity`" is displayed instead of the actual UB value. Similarly, if LB is equal to the negative of the maximum storable value of an `int`, then "`-infinity`" is displayed instead of the actual LB value.
- `ERROR: Input must be a real number in [<LB>, <UB>]!`
  where `LB` and `UB` are lower and upper bounds. If UB is equal to the maximum storable value of a `double`, then "`infinity`" is displayed instead of the actual UB value. Similarly, if LB is equal to the negative of the maximum storable value of a `double`, then "`-infinity`" is displayed instead of the actual LB value.

4 Required elements

Your project must use the each of the elements described in this section. You can have more classes, methods (functions), variables, and objects if you want, but you must at least use the elements described in this section. **Failure to correctly implement any of the elements in this section may result in a zero on the project.**

Variable and method prototypes and short descriptions are provided below. It is your job to determine exactly what should happen inside each method, though it should be clear from the function and argument names and function descriptions.

class `FeetInches`: A helper class for storing both feet and inches and supporting conversions.

- `static final IN_TO_CM`
  Global constant equal to 2.54.
- `private double _feet`
  Stores the number of feet for this class instance.
- `private double _inches`
  Stores the number of inches for this class instance.
- `public FeetInches(double feet, double inches)`
  A constructor that initializes the class members when `new FeetInches(...)` is called.
- `public double getFeet()`
  Retrieves the number of feet stored in this instance.
- `public double getInches()`
  Retrieves the number of inches stored in this instance.
- `public String toString()`
  Allows the class instance to be converted to a String, which allows printing by `System.out.print(...)`. You should define this method, but you do not have to use it if you prefer not to.
- `public double convertToCm()`
  Converts the feet and inches stored in this class to centimeters.
- `public static FeetInches convertToIn(double cm)`
  Returns a new instance of FeetInches representing the conversion of the parameter (in centimeters). This is static because it does not need to access any class members of instances – everything it needs to know is passed in the parameters. Because this method is static, it has to create a new instance of FeetInches to represent the result of the conversion and return it.

class `Pro2_utorid`: The class which executes the main control flow.

- `public static BufferedReader` as the **the only** global `BufferedReader` object in the entire program for reading user input.
  - Although your programs will compile and run just fine on your own computer with multiple `BufferedReader` objects, they will not run correctly on a web server with multiple `BufferedReader` objects. Since your programs will be graded on a web server, please don't have more than one `BufferedReader` for reading input from the console.
- `public static String[] _prevConversions = new String[1];`
  Stores the String representing the output of the conversions previously made. Whenever the size of this array needs to increase, the size should be doubled.
- `public static int _numConversions = 0;`
  Used to keep track of the number of conversions currently stored in `_prevConversions`.

- `public static void displayMenu()`
  Display the menu.
- `public static String feetInchesToCm()`
  Get feet and inches from the user, then return the String showing the number of centimeters.
- `public static String cmTofeetInches()`
  Get centimeters from the user, then return the String showing the number of feet and inches.
  - The two methods above return Strings so that these Strings can be printed immediately as well as saved in the `_prevConversions` array by calling `addConversion(...)` below.
- `public static int getInteger(String prompt, int LB, int UB)`
  Get an integer in the range [LB, UB] from the user. Prompt the user repeatedly until a valid value is entered.
- `public static double getDouble(String prompt, double LB, double UB)`
  Get a real number in the range [LB, UB] from the user. Prompt the user repeatedly until a valid value is entered.
- `public static void addConversion(String s)`
  Adds the String representing the output of the current conversion to the `_prevConversions` array and increments `_numConversions`.

5 Helpful hints

- Remember, you will be graded on whether or not your program produces **EXACTLY** the same output as the provided solution program—including spaces, spelling, punctuation—everything counts! So, to make sure your output is identical to the solution output, copy and paste segments from the solution program (e.g., the menu) directly into your own source code.
- Test everything. What happens if you enter a negative number somewhere? Or a letter instead of a number? Or a number instead of a letter? Or a decimal number instead of an integer?
- If a particular action makes the solution program crash (e.g., entering a letter instead of a number), then your program can crash, too. If the solution program does not have a particular error check, then your program does not need to have it either. Think of it as a freebie.
- The maximum value that a `double` variable can take on is `Double.MAX_VALUE`.
- The maximum value that an `int` variable can take on is `Integer.MAX_VALUE`.