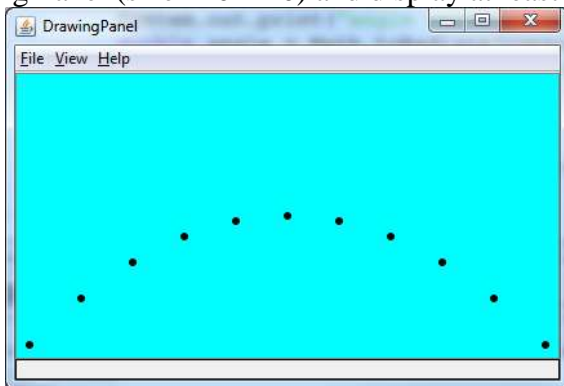


READ ALL INSTRUCTIONS BEFORE STARTING YOUR SOLUTION.

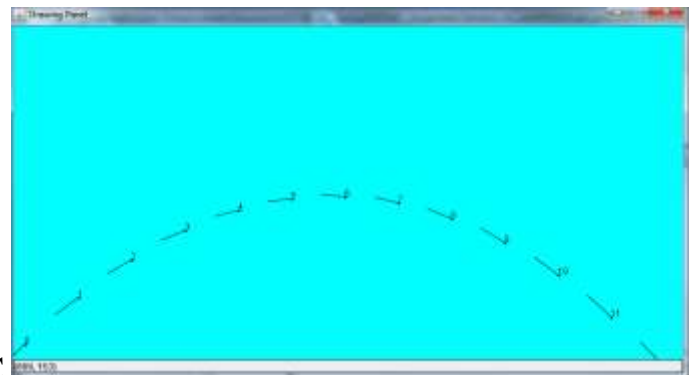
Topics: ch. 4-5:for loops, ifs, random numbers, and using object analysis, structured, top-down design, methods, expressions, variables, loops, class constants, formatting, Graphics, interactive programming and more analysis, structured, top-down **design**, expressions, variables, loops, class constants, formatting, Graphics, interactive programming (user input), selection (`if`), iteration (`for`), variables, methods, programming style, integer issues and basic analytic geometry.

Specifications:

1. **You must follow all the style requirements in *Java_Program_Style_Requirements_GradingComponents* to earn full credit.** That document describes the scoring process used and some design hints.
2. ****You must use the graphics package stored in the file *DrawingPanel.java*. You must put a copy of this file in the folder with your code and compile it before compiling your code.**
3. **Part Sp6A**yourInitials_Projectile.java: Start with the chapter 3.4 case study: projectile trajectory, pp. 168.
4. Refine and re-structure the existing code to meet ALL the CS141 programming Style Requirements
5. Remove redundant code and ensure each input is isolated in one or more methods, each output is isolated one or more methods. Ensure, as much as possible, each set of calculations is isolated in one or more methods.
6. Build helper methods for getting valid input (add loop and if code to get reasonable input values.)
For example, you should create: `int getInt(String prompt, int max, int min, int default) { }`
7. At the top of your console output, display author, title and a brief introduction; but, no offensive remarks.
8. You may assume valid user input **types**. When the user is prompted for numbers, you may assume the user will type integers, **perhaps not within proper ranges**.
9. You must structure your solution using static methods that accept parameters and return values where needed.
10. You must have **at least 3 non-trivial methods other than main** in your program – see the next specification.
11. You must define other methods as needed for structure and to eliminate redundancy.
12. Build helper methods to do the physics formulas and simplify the control structure
13. **Test then print that java listing**, display panel and console output for 30 m/s, 45 degrees, 10 steps to **hand in**.
14. **Part Sp6B**yourInitials Add the graphic display of the projectile path of a small black cannon ball. Use Drawing Panel (size 420x220) and display at least 10 positions on the arc. (See #4, p. 205 of the first edition.)



SP6B



SP6C

15. **Test then print that java listing**, display panel and console output for 30 m/s, 45 degrees, 10 steps to **hand in**.
16. Email that properly named .java file with the cannon ball projectile path along with your part C code.
17. **Part Sp6C**yourInitials_Javelin.java: Change the cannon ball to a javelin. You do not need to print the step numbers with the javelin. The image above was used for debugging purposes.
18. Adjust the angle of the javelin throughout its flight to **approximate** reality.
Test then print that java listing, display panel and console output for 30 m/s, 45 degrees, 10 steps to **hand in**.
19. In addition to 3 or more designs, for part B and C, you must turn in clearly identified and stapled (in order) :
 - (1) Your paper .java code listing with **line numbers** and
 - (2) Screen-captures of the console window showing the required example inputs listed above and
 - (3) the drawing panels showing the required example outputs listed above.