Final Design Project

1. Expectations:
    a. This is an individual project, which will score 125 points. You are free to use your textbook, notes, items found on the Internet. You may ask me questions.
    b. Unlike the May AP test, you may use Eclipse. It will help you by identifying compile time errors.
    c. In particular, you may not receive help from classmates on the project. This constraint is intended to help you with the FRQ portion of the May AP test. On that test, you need to be able to write code, without an IDE or help, that is free of compile time, run time, and logic errors. Eclipse will identify these errors for you, however at this stage, it is critical that you at least discover (by yourself) the causes of those errors. If you ask for help from a classmate, it will be treated in the same fashion as if you asked for help during a math\science\English test.
    d. There are two rubrics on Canvas; one for the coding\debug, one for the presentation

2. PROJECT: Life has been discovered on Mars. At present, the number of life forms is significantly different than those found on Earth. On Earth, the biologic classification system is organized as Kingdom-Phylum-Class-Order-Family-Genus-Species. Mars will be different. The scheme will be Kingdom-Motives-Ingesting-Genus-Species, as follows:
    a. Kingdom: Latin word Martis.
    b. Motives: floating, crawling, burrowing, and sedentary. Corresponding Latin words would be pendens, repens, perfodiens, and sedens. Both Latin and English words should be object instance variables.
    c. Ingesting: gaseous, carnivorous, processing. Corresponding Latin words would be ingerens vaporem, ingerens carnem, and ingerens perficiendo. Both Latin and English words should be object instance variables.
    d. Genus: your choice of word
    e. Species: your choice of word, you may name a life-form after yourself

3. Your program will include:
    a. Parent class = Kingdom
    b. Child class = Motives, extension of Kingdom

   c. Child class of Motives = Ingesting, ingesting is an extension of Motives class

4. Kingdom class:
   a. Constructor
   b. toString
   c. Instance variables of your choice

5. Motives class, an extension of Kingdom, will have
   a. Constructor
   b. toString
   c. Instance variable value that indicates pendens, repens, perfodiens or sedens type
   d. Instance variables of your choice
   e. compareTo method

6. Ingesting class, an extension of Motives, will have
   a. Constructor
   b. toString
   c. Instance variable value that indicates vaporem, carnem, perficiendo
   d. Instance variables for Genus, Species
   e. Other instance variables of your choice.
   f. compareTo method
   g. Normal population per square mile

7. You will create and load an array of 20 Ingesting objects, as directed by prompts from the user (see paragraph 9.a). To avoid tedious and continuous data entry, you may create 17 of these without user prompts; i.e. call constructors with fixed values. Three objects must be created using values obtained from a user, and menu choices provided by the program.

8. Main calls a menu function, initializes key variables and constants, other management functions of your choice.

NOTE: Menu function(s) must not crash in response to basic user entry errors. For instance, if a user is to choose an option between 1 and 9, and they enter "0". For yes/no answers, must accept Yes, YES, yes; and No, NO, no.

9. Menu function presents options for the following. It is recommended that these be individual methods, called by a switch\case statement
    a. Loading initial array. This method presents users with the "Motives" choices listed in paragraph 1.b. Then presents users with the "Ingesting" choices listed in paragraph 1.c. Prompts for Genus and Species. It then uses Ingesting constructor to create ingesting objects and load the array of 20 objects. Ingesting constructor should call Motives constructor with appropriate paragraph 1.b. information.
    b. Printing genus and species of array objects. Need to use toString, which should call toString of parent class.
    c. Print total classification for an object in the array. This means the objects Kingdom-Motives-Ingesting-Genus and Species values
    d. Compare genus-species values in the array, determine if they are the same Motives-Ingesting type or not.
    e. Decide if a newly discovered lifeform is already in the array; does not have to have same genus-species variable values, comparison would be based on motives and ingesting values. If lifeform is not in the array, suggest a Motives-Ingesting type for a new life form
    f. Estimating population of each item in the array in a user specified area; i.e. user states a 500 square kilometer area, function returns the expected population of each life form in that area.
    g. Estimating population at a user specified future time. Use a non-linear math function to predict future population (see paragraph 14.c). NOTE: your growth algorithm may assume that population increases, or that it decreases. Algorithm must be commented (see paragraph 15.d)
    h. Exit menu. Menu repeats otherwise
    i. EXTRA CREDIT: permit user to see and change selected object instance variables
    j. EXTRA CREDIT: create graphic of a few life forms. Does not need to be done inside of JAVA.

10. You need to define life form instance variables that include some numeric values, i.e. lifespan, size, mass.

11. Be creative; define parent instance variables, child class instance variables. If you are defining the floating life form (pendens), you might consider instance variables that indicate if the organism floats at high altitude or low, if it floats only in Polar Regions, if it can change

altitude\direction. These could play into your Genus and Species choices.

12. For ingerens vaporem, specify the gases that the life form ingests\processes. Use only gases present in Martian atmosphere

13. For ingerens carnem, consider stating which organisms it eats (which could be all organisms)

14. Must use at least one method from each of the following tables
    a. Page 78, Figure 2.5, Strings
    b. Page 85, Figure 2.11, Random
    c. Page 88, Figure 2.12, Math
    d. Page 90, Figure 2.13, Scanner
    e. Page 95, Figure 2.15, DecimalFormat
    f. Page 264, Figure 5.4, compareTo

15. Comments:
    a. Each method is preceded by a multi-line description of the method
    b. Each closing bracket, comment identifies beginning bracket (i.e. // end of while loop), unless the entire block of code is less than 5 lines of code
    c. Version control at top of file; minimum is your name, date of last edit, and brief description of the file.
    d. Algorithms should be preceded by a comment that describes them.