P0

Programming Assignment 0: Welcome to / Review Java

Specification

Overview

This assignment is intended to warm up coding in CSE 122 with the Java programming language. Depending on your programming experience, this assignment might feel a little different for you.

- *For students who took CSE 121*: Welcome! 🎉 This assignment should review the concepts and programming skills you learned in CSE 121.
- *For everyone else:* Welcome! 🎉 This assignment should review the fundamental concepts of programming (variables, looping, conditionals, methods/functions, arrays) that you should be familiar with from your prior programming experience. If you are coming from a programming language other than Java, this assignment will allow you to practice these familiar programming concepts in Java. See the Java Tutorial for examples of these fundamental coding constructs in Java.

Learning Objectives

By completing this assignment, students will be able to:

- Predict the behavior and results of executing Java code that uses basic control structures and data abstractions
- Write functionally correct Java programs that meet a provided specification using control structures, primitive data types, and basic data abstractions
- Write programs and classes that are readable and maintainable, and that conform to provided guidelines for style and implementation

printTriangle - Spec

This slide contains the specification for `printTriangle`. Write your solution in the `Triangle.java` class in the **Coding Workspace** slide.

Skills: Printing, Loops, Methods, Parameters

Write a method called `printTriangle` that takes an integer `numStars` and prints a triangle with two sides of length `numStars`. For example, `printTriangle(5)` would produce the following output:

- ****
  ****
  ***
  **
  *

As another example, `printTriangle(3)` would print

- **
  **
  *

You can assume that the number `numStars` is greater than 1.

Commenting & Code Quality

- `printTriangle` is intended to be a warm-up problem and is only graded on Behavior and not on Code Quality or Commenting. You are still encouraged to practice writing code with good Code Quality and with comments, but it is not a requirement for the `printTriangle` method.

Census - Spec

This slide contains the specification for Census program. Write your solution in the main method of `Census.java` class in the **Coding Workspace** slide. This file starts out blank so you will need to write the class definition and the main method yourself!

Skills: Printing, Variables, Expressions, Loops, Conditionals, User Input

Context: Every 10 years, the United States conducts a census to get an up-to-date representation of demographics in country. To help the Census workers collect and understand the data, they recruited you to help them write a program to analyze the ages of the US population.

***Task**: Write* a complete runnable Java program named `Census` (including the class definition and main method) that lets a Census worker type in a sequence of ages for each person they collected data from, and then prints out the average of all of the ages inputted.

The program should repeatedly ask for an age of the next person until the Census worker types in a negative number, after which it will print the average age of all of the ages previously typed in.

The following text shows one example log of the program's output. The **bold underlined text** shows the user input and does not need to be printed by your program as that is what the user will type into the computer.

Welcome to the Census!
Input the ages of the population and we will compute the average age
Next person's age (negative to quit)?
14
Next person's age (negative to quit)?
40
Next person's age (negative to quit)?
39
Next person's age (negative to quit)?
7
Next person's age (negative to quit)?
0
Next person's age (negative to quit)?
-4
The average age is 20.0

Another example interaction is shown below.

Welcome to the Census!
Input the ages of the population and we will compute the average age
Next person's age (negative to quit)?
27
Next person's age (negative to quit)?
10
Next person's age (negative to quit)?
13

Next person's age (negative to quit)?
-1
The average age is 16.666666666666668

You can make the following assumptions:
• You should use the
`Scanner` class to get user input.
• The user will type in at least one non-negative number.
• The user will always type in integer numbers and you do not need to handle invalid input. Zero is considered a
valid number for this program.
• You do not need to worry about rounding the output of the average. For example, it is possible for the program to print out an average of
`3.3333333333333335`. You can just print the result of the average without doing anything special to the output.
Commenting & Code Quality

• Each method in your program (other than
`main`) should have a comment describing its behavior, its parameters, and its return (if non-void). Your method comments should also avoid mentioning any implementation details.
  ◦ Refer to the
CSE 122 Commenting Guide for guidance on how to write good method comments!
• Your code should otherwise adhere to the code quality guidelines outlined in the CSE 122 Code Quality Guide. In particular, you should use descriptive names for methods and variables, follow naming conventions, use correct and consistent indentation, and utilize programming constructs appropriately.

mostCommonNucleotide - Spec

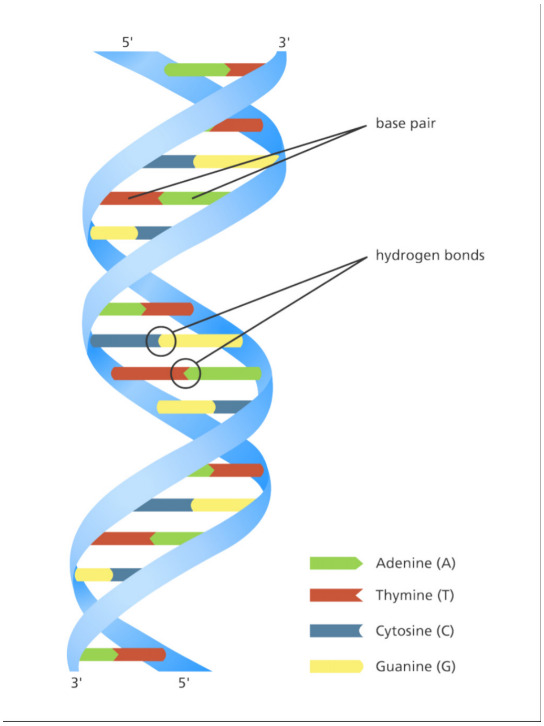
This slide contains the specification for `mostCommonNucleotide` method. Write your solution in the main method of `DNA.java` class in the **Coding Workspace** slide.

Skills: Loops, Conditionals, Strings, Methods, Parameters, Returns, Arrays, Commenting
Background

Note: This section explains some information from the field of biology that is related to this problem. It is for your information only; you do not need to fully understand it to complete the assignment.

Deoxyribonucleic acid (DNA) is a complex biochemical macromolecule that carries the genetic information for cellular life forms and some viruses. DNA is also the mechanism through which genetic information from parents is passed on during reproduction. DNA consists of long chains of chemical compounds called **nucleotides**. Four nucleotides are present in DNA:

- Adenine (A)
- Cytosine (C)
- Guanine (G)
- Thymine (T).



DNA has a double-helix structure (see diagram below) containing complementary chains of these four nucleotides connected by hydrogen bonds.

DNA's double helix structure is useful for replication and checking the consistency of the genetic code. However, when analyzing DNA we generally only need the nucleotides from one of the strands. Only certain nucleotides can be matched on opposing sides of the double helix, so knowing the nucleotide in one strand is sufficient for the entire genetic code.
In other words, the other strand is a sort of duplicate of the first as it contains the same information.

Your Task

For this problem, we will consider a String encoding DNA information as a sequence of As, Cs, Gs, or Ts. For example, we could represent a sequence of DNA as the String `"ATGCGCACTATGGTAG"` or `"TAGC"`.
Write a method called `mostCommonNucleotide` that takes a DNA string as a parameter and returns the nucleotide that appears most frequently in that DNA as a string. Here are some examples of how this method may be called and what it should return.

- `mostCommonNucleotide("ATGCGCACTATGGTAG")` should return `"G"`.
- `mostCommonNucleotide("GAAAGAAA")` should return `"A"`
- `mostCommonNucleotide("T")` should return `"T"`.
- `mostCommonNucleotide("TAGC")` should return `"A"` (see the next paragraph for an explanation of why).

In the case that two nucleotides are tied for appearing the most frequently, you should return the nucleotide that comes first in the order of A, C, G, T. So, for example, if there is a tie between C and T, you should return C because it appears before T in this sequence A, C, G, T.
You may make the following assumptions about the input:
• The provided string is non-empty and contains only uppercase letters A, C, G, or T.

Implementation Guidelines

• You should use

functional decomposition to break `mostCommonNucleotide` into at least two additional helper methods to solve non-trivial sub-problems of the task. See the example from class this week on how to decompose larger problems into smaller ones.

-

Hint: Think of how you would describe your algorithm (i.e. the sequence of steps needed) to solve your problem. If you find yourself saying, "To solve this, I need to do A and then B", A and B might be good candidates for helper methods!

- Note that you are being asked to write two helper methods *in addition to* `mostCommonNucleotide` so your code should have at least 3 methods total in addition to `main`.

• Your solution should use an `int` array of length 4 to represent the counts of the nucleotides A, C, G, and T. Note that you will likely want to choose which letter belongs in which index carefully to handle the tie-breaking case. *Hint: It is okay to use if/else statements to decide how a letter gets put into a certain index, but there are also other ways to solve this task.*

Reflection

The following questions will ask you practice **metacognition** to reflect on your process of completing this assignment. For each question, focus on your plan and/or process for working through the assignment rather than the CS concepts. Think about things like how you organized your working time, what sorts of things tended to go wrong, and how you dealt with those errors or mistakes.

Question 1

What did you learn or what skills did you practice on this assignment?

I reviewed functional decomposition by breaking down the problem into smaller problems and creating helper methods to solve the smaller problems. I also practiced commenting/documenting. This gave me an opportunity to practice explaining the code and also helped me better understand what my own program does. I also reviewed topics such as variables, objects, primitive data types, methods/functions, parameters, returns, control structures, and loops.

Question 2

What did you find challenging or struggle most with on this assignment? How did you

work to overcome that struggle (even if you couldn't yet)?
I struggled breaking down some of the problems into smaller problems and creating helper methods to solve those problems. To try and solve this I wrote down an initial plan on paper explaining the various parts of my solution before actually programming it. I also struggled with explaining what my code does but by working on writing the comments I now have a better understanding of my code and how to explain my program.

Question 3 Submitted
How will you use what you learned from this assignment to improve your process or approach on future assignments (e.g., review specific topics, create a cheat sheet with Java syntax)?
I will use this assignment as a guideline for what topics I should review. For instance I struggled with functional decomposition so I will make sure to practice functional decomposition, such that I master it. I will also note down useful patterns that I found while solving the problems as well as the steps I took to solve each problem that way I have an idea of how to go about solving problems I encounter later on as well.

Question 4
Explain your reasoning for how you decided to use *functional decomposition* to break down `mostCommonNucleotide` into smaller problems.
I decided to use functional decomposition to break down the mostCommonNucleotide into smaller problems because I noticed how when coming up with a plan for the program my solution essentially consisted of two parts. One part of my solution consisted of counting the number of each type of nucleotide and storing them in an array. Another part of my solution consisted of taking that array containing the counts of each type of nucleotide and finding the index in the array that has the maximum count, such that I would be able to use that index to return the corresponding String that is the most common nucleotide.