JAVA DELIVERABLE 2

Message Encoder

Did you ever send coded messages as a kid? Replace A with 1, B with 2, etc., all the way through Z = 26. In your second deliverable of Lab 1, you're going to write a program to do just that.

The same information we shared in Deliverable 1 apply:

Disclaimer**:** A large part of being a developer is researching and understanding new mechanics and concepts of coding. Every developer, even a seasoned veteran, needs to look up and research coding concepts. As such, for this exercise, you may need to do some research.

Here are a few hints:

- For any programming language, Google and Stack Overflow will be your go-to sites for learning about code.
- Google is good at answering common questions, Stack Overflow is good for troubleshooting and reading issues other programmers have encountered.

Put this project in its own repo on GitHub and submit the GitHub link in the Turn In Deliverable 2 spot in the LMS.

MESSAGE ENCODER

Task: Write a program that takes a word from the user and converts it to a basic code: A is 1, B is 2, etc. Output the numbers with dashes between them.

Build Specifications:

- Declare three variables:
  - **input** will hold the user's string input from the console
  - **message** will hold the encoded string (because there are multiple numbers, it should still hold a string)
  - **checksum** will hold the total Unicode value of the string
- Convert the user's input to all upper case before encoding.
- Go through the string and find the Unicode value of each letter in turn.
- Add the Unicode value for each letter into the checksum
- Add the encoded letter to the message string, followed by a dash. (It's OK if you have an extra dash at the end.)
- Output the message and the checksum.

Assume the user is only entering letters. You don't need to worry about numbers, symbols, spaces, or anything but letters.

HINT #1: You do not need to write an if statement with 26 conditions nor a switch statement with 26 cases.

HINT #2: In Unicode (and ASCII), a capital A has a value of 65.

Example run (user input in bold):

```
What is your message? Java

Your encoded message is 10-1-22-1
Message checksum is 290
```

Explanation of run:

- J = 10, A = 1, V = 22, A = 1 This gives the message `10-1-22-1`. (Note that it is also acceptable to have an extra dash, like this: `10-1-22-1-`.)
- The checksum adds the *Unicode* number for each letter. J = 74, A = 65, V = 86, A = 65. 74 + 65 + 86 + 65 = 290.

Grading Rubric: This is graded out of 10 points. You must score 8 or more points on each deliverable in Lab 1 to pass.

1 point each. No partial credit is allowed on an individual point. Credit will be granted for any points that are written correctly themselves, but don't run correctly because of a problem elsewhere in the program.

1. Correctly gets user input from the console
2. Stores user input in a variable named **input**.
3. Converts user input into all caps.
4. Loops through the input string
5. Loops correctly--starts and ends at the right place
6. Gets the Unicode value for an individual letter
7. Shifts the Unicode value to "code" correctly (A is 1, B is 2, etc.)
8. Stores the coded message in a variable named **message** and concatenates correctly.
9. Sums the Unicode values in a variable named **checksum**.
10. Outputs the message and checksum correctly.

Grading Scale:

8 or above Passing
Below 8: Not Passing