Strings Review

Write programs that accomplish each task. Use appropriate variable names and prompts for user input when necessary. Include a header for each *program* and for each *function*. Ensure that all values input by the user are within acceptable ranges. Do *not* assume that input is the correct type: write code to validate user input.

1. Read an arbitrary number of integers from the user and determine their average. Your program should not crash given invalid input, or given an inappropriate number of values to average.

2. Read a string consisting entirely of letters from the user, then determine its "score" using the following rules:
   - each letter has a value determined by its position in the alphabet (e.g. A=1, Z=26)
   - each letter's value is multiplied by the value of its position in the string (e.g. second letter x2)

   For example, the word "dog" would have a score of 4x1 + 15x2 + 7x3 = 55. Your program should handle both lower- and uppercase input.

3. *Word Guess*: In this two-player game, one player chooses a secret word which the other has to guess by entering all letters. The player also provides the category as a hint. For example, if the secret word is "macaroni" the category might be "food". The secret word is displayed as a series of blanks, and as each letter is guessed, *all* instances of that letter are revealed. Thus, "macaroni" would be displayed as _ _ _ _ _ _ _ _, and if the player guesses the letter 'a', it would become _ a _ a _ _ _ _. If the guesser makes six mistakes, they lose the game and the secret word is revealed.

   Each round, you should display the current status of the word (where correctly-guessed letters are filled in), the number of allowable mistakes remaining, and a listing of all letters that have not been guessed. For each guess, the listing of letters should be updated with that letter removed as an option. If the player tries to guess a previously-guessed letter, inform the user but do not penalize them by deducting a guess. If the player successfully guesses the word, display a winning message and end the game. If they lose, the game should end with a losing message. Below is an example of how this might look.

```
CATEGORY: TOYS
_ _ _ _         Guesses left: 6   Avail. letters: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Guess a letter: E
There is no E in the word.
_ _ _ _         Guesses left: 5   Avail. letters: ABCD FGHIJKLMNOPQRSTUVWXYZ
Guess a letter: O
_ O _ _         Guesses left: 5   Avail. letters: ABCD FGHIJKLMN PQRSTUVWXYZ
Guess a letter: E
You've already guessed that letter!
Guess a letter: T
There is no T in the word.
_ O _ _         Guesses left: 4   Avail. letters: ABCD FGHIJKLMN PQRS UVWXYZ
Guess a letter: L
_ O L L         Guesses left: 4   Avail. letters: ABCD FGHIJK MN PQRS UVWXYZ
Guess a letter: D
Congratulations! The word was DOLL.
```