CS101 Lab: Building a Hangman Game in C++

# Lab Title:

Hangman with Strings

# Lab Objective:

In this lab, you will write a C++ program to implement a simple version of the classic game Hangman. This project will help you practice using string methods and working with loops, conditionals, and functions in C++.  
  
By the end of the lab, you should be comfortable:  
- Using string methods like .length(), .find(), .substr(), .replace(), etc.  
- Using loops and conditional statements  
- Reading and validating user input  
- Manipulating and comparing strings

# Background:

Hangman is a word guessing game. The program picks a secret word, and the player tries to guess it one letter at a time. The player has a limited number of incorrect guesses. If the player reveals all letters before running out of guesses, they win!

# Requirements:

1. Choose a word:  
 The program should use a hard-coded word (e.g., "computer"), or randomly select from a list.  
  
2. Hide the word:  
 Display the word as underscores (\_) for each letter (e.g., "apple" → "\_\_\_\_\_").  
  
3. Take user guesses:  
 Prompt the user to guess one letter at a time. Update the display if correct.  
  
4. Validate Input:  
 Ensure the user enters only a single alphabetic character.  
 Example: 'a' is valid; 'ab', '1', or '%' are not.  
  
5. Track incorrect guesses:  
 Allow up to 5 incorrect guesses.  
  
6. Win/Lose Logic:  
 The user wins if they guess all letters before 5 incorrect guesses.  
 The user loses otherwise.  
  
7. Endgame Message:  
 Display whether the user won or lost and reveal the full word.

# String Methods to Use:

You must demonstrate use of at least three of the following string methods:  
- length()  
- find()  
- replace()  
- at() or operator[]

# Sample Output:

Welcome to Hangman!

The word is: \_ \_ \_ \_ \_ \_ \_  
Guess a letter: e  
Correct!

The word is: \_ \_ \_ \_ \_ \_ e  
Guess a letter: 5  
Invalid input.

Please enter a single letter.  
Guess a letter: a  
Sorry, no 'a'.

You have 1/5 incorrect guesses.  
...  
You won! The word was: "hangman"

# Suggestions / Tips:

- Use a function to update the displayed word after each guess.  
- Normalize input using tolower() or toupper().  
- Store the current guessed state in a separate string.  
- Use a while loop until win/loss condition.  
- Track previously guessed letters.

# Stretch Goals (Optional):

- Allow guessing the full word.  
- Allow user to win points/”money” for each letter they get correct and more for each word they get correct. The user can then buy more guesses when they run out.  
- Draw ASCII hangman figure.