Aror University of Art, Architecture, Design & Heritage Sukkur.

Department of Artificial Intelligence and Multimedia Gaming

Programming for AI (Spring-2025)

LAB#03

Prepared by: Abdul Haseeb Shaikh

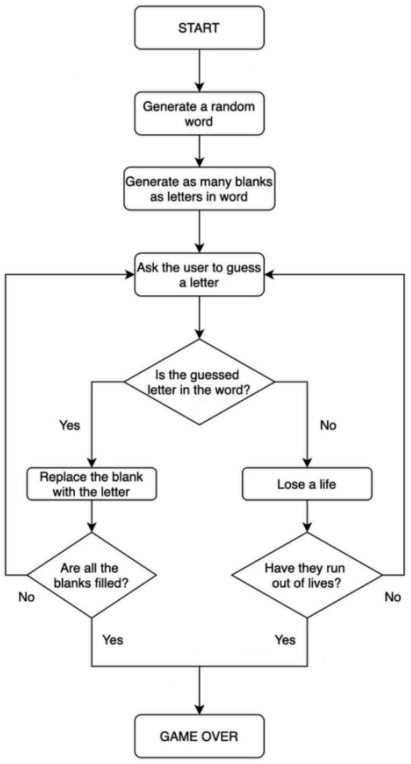
Objective of Lab No. 3:

After performing lab3, students will be able to:

- Implement while loop
- Implement Functions
- Implement Dictionaries in python

1. Create a Hangman game in python, the flowchart for the game is shown below:

Aror University of Art, Architecture, Design & Heritage Sukkur.



Hints:

1. Create a List of Words
2. Pick a Random word from the list
3. Generate as many _ as letters in the chosen word
4. Ask the user to guess a word, if the guess is in the chosen word replace the blank with that word, continue the process until the user guesses all the letters in the chosen word or the game is over (when lives become zero)
5. Remember user has 10 lives, on each wrong guess one life is deducted

2. **Write down a function to check whether a given number is a prime number or not, remember a prime number is only divisible by itself and 1.**

3. **Create a Ceaser Cipher Program:**

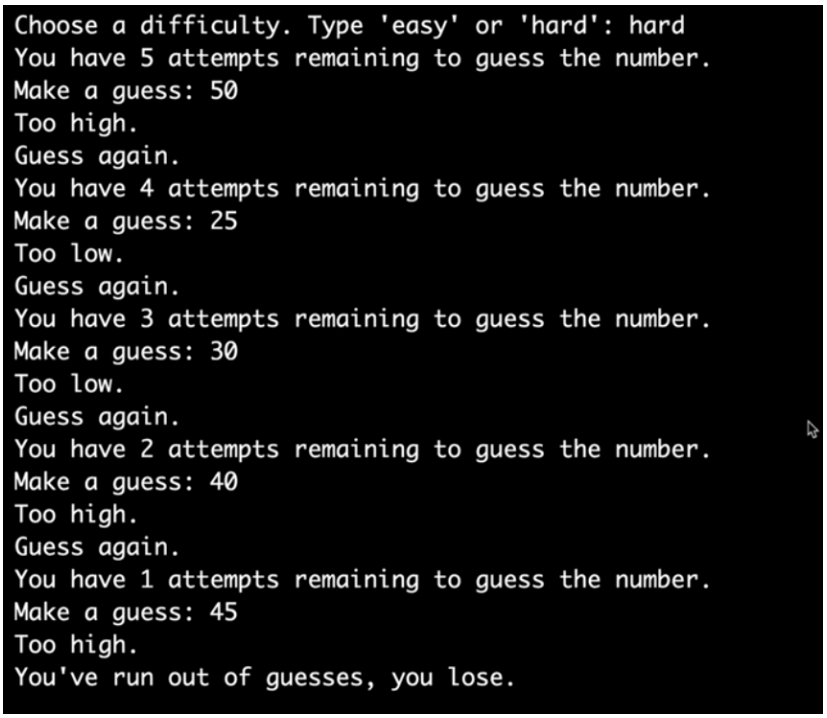
    a. Create a function called encrypt which takes one string, and one shift which is basically an integer number as input, and it encodes the given input

For example: if input text was hello and shift was 5, then encoded output will be mjqqt

b. Create a function called decrypt which decoded the encoded input, provided with same shift, suppose if the user gave mjqqt as input and shift was 5, then output will be hello.

4. **Create Secret Auction Program:**
   a. The program asks the user about his/her name and the bid amount, after that the program asks if there are any other Bidders (yes/no)
   b. If there are other Bidders, the program clears the screen using (from replit import clear)
   c. This process continues until there are no other bidders
   d. Finally, the highest bidder wins

5. **Implement the Number Guessing Game as shown in the below Diagram:**



6. Employee Performance Tracker:

a. You are given a dictionary where each key is an **employee name**, and the value is another dictionary containing:

b. "tasks_completed" (list of task completion times in hours)
c. "department" (e.g., "HR", "IT", "Finance")
d. Your task is to:
e. **Calculate the average task completion time** for each employee.
f. **Categorize employees based on performance:**
g. **Excellent:** Avg time ≤ 2 hours
h. **Good:** Avg time > 2 and ≤ 4 hours
i. **Needs Improvement:** Avg time > 4 hours
j. **Find the best-performing department**, based on the highest percentage of "Excellent" employees.
k. **Find the slowest employee overall**