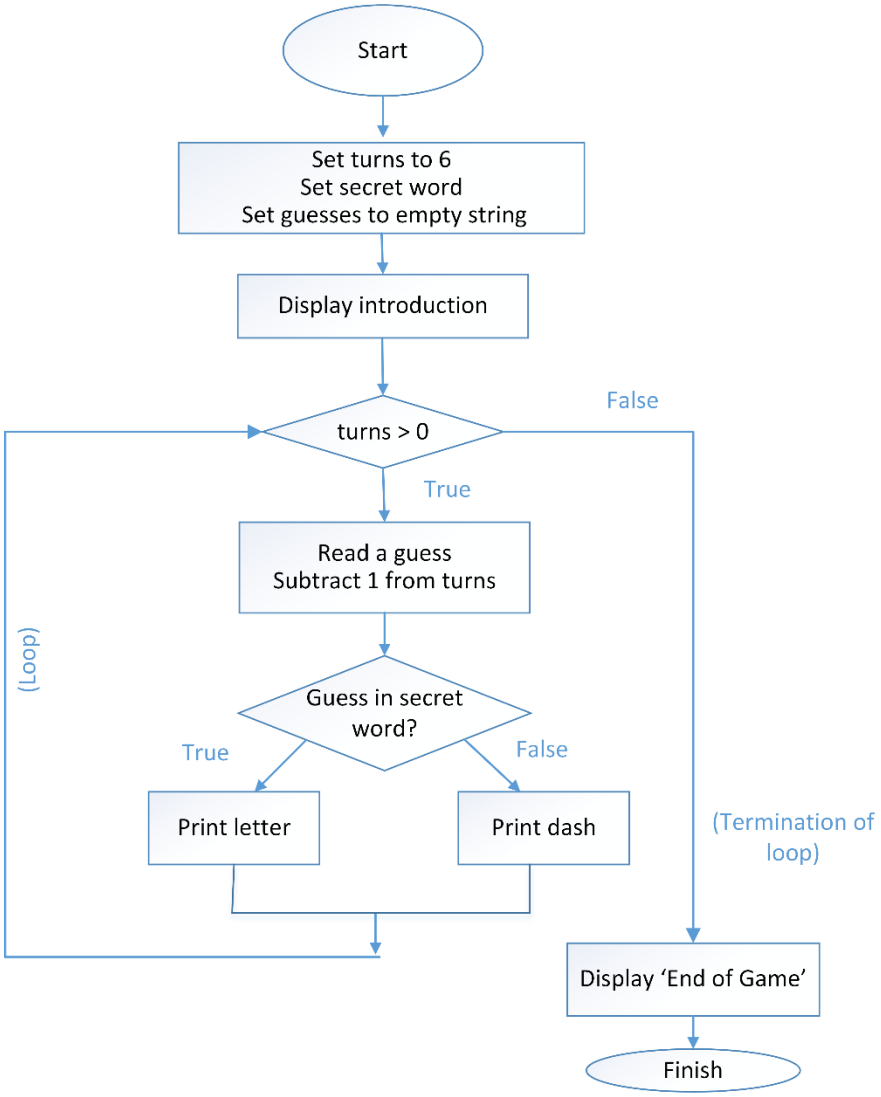
Week 5 Tutorial – Guess the Word

The program created for this tutorial will use *flowcharts*, *strings*, *for* loops, *while* loops and *if-else* conditions.

Create a python program to play 'Guess the Word'.

- The program will store a secret word (variable *secret*)
- The player will guess a letter in the word (variable *guess*).
- The program will store each guess entered by the player (variable *guesses*).
- The player will be allowed a number of guesses (variable *turns*).
- The program will print a dash (_) for each letter in the secret word not yet found.
- If the players guess is found in the word, the program will display the letter in replace of the appropriate dash.
- The program will display appropriate introduction and end of game messages.

Sample Flowchart – <u>This relates to exercises 1, 2 and 3 only.</u> You should work out how to design and implement exercise 4 and the additional exercises.



1. Getting started (shown on flowchart)
   - Store the string 'westminster' in a variable called *secret*.
   - Set variable *turns* to 6 (subtract 1 from turns each time the player takes a guess).
   - Store an empty string in a variable called guesses (to store a string of all the guesses entered by the player).
   - Display introduction:
     - Print a message that announces the start of the guess the word game.
     - Print the number of turns the player has.
     - Print the letters in the secret word replaced by a dash (underscore).
       - **Hint:** In Python the * operator can be used to performs repetition on strings. Therefore, you can print ' _ ' multiplied by the length of the string

   - Example output:

```
Let's play Guess the Word
You have 6 turns to guess the word!
 _  _  _  _  _  _  _  _  _  _  _
```

   - Save and run the program to check that it works as expected.

2. Ask player to guess a letter (shown on flowchart)
   A) Update the program to let the player enter a letter and store it in a variable named *guess*. Then concatenate (join) the guess to the *guesses* variable.
   B) The program should show where any guessed letters are. Use a *for* loop to traverse the string variable *secret*. Examples:
      - The following uses a *for* loop to print each letter of a string on the same line:

```
for letter in secret:
    print('', letter, '', end='')
```

      - The following prints a dash (underscore) in replace of each letter of a string.

```
for letter in secret:
    print(' _ ', end='')
```

      - Use the above snippets to create a *for* loop that will traverse the secret word letter by letter. For each loop, check if the letter in the secret word is stored in variable *guesses*.
        - If it is in *guesses*, print the letter.
        - Otherwise, print a dash (underscore)
      - Add an end of game message.
      - Example output (user input in bold):

```
Let's play Guess the Word
You have 6 turns to guess the word!
 _  _  _  _  _  _  _  _  _  _  _
Guess a letter: w
w  _  _  _  _  _  _  _  _  _  _
End of Game
```

- Remember to use the correct indentation for the loops and conditions.
- Save and run file. The program should allow the user to guess a letter and show where it is in the string if it is found.

3) Allow the player to take multiple guesses (shown on flowchart)
Use a while loop to repeat the program until the player is out of turns.

- Use the value in variable *turns* to control the while loop.
  - Earlier you created a variable called *turns* and set it to 6.
  - Add a while loop with a condition that checks if *turns* is above zero.
  - Within the loop subtract one from *turns*.
  - When turns is finally zero, the "while" will stop repeating.

Example output (user input in bold):

```
Let's play Guess the Word
You have 6 turns to guess the word!
 _  _  _  _  _  _  _  _  _  _  _
Guess a letter: e
 _  e  _  _  _  _  _  _  _  e  _
Guess a letter: i
 _  e  _  _  _  i  _  _  _  e  _
Guess a letter: w
 w  e  _  _  _  i  _  _  _  e  _
Guess a letter: x
 w  e  _  _  _  i  _  _  _  e  _
Guess a letter: y
 w  e  _  _  _  i  _  _  _  e  _
Guess a letter: z
 w  e  _  _  _  i  _  _  _  e  _

End of Game
```

- Run your program and test that it works.

4) Player should wins when word is guessed (NOT shown on flowchart)
The current flowchart and program show the program loops until the number of turns reaches zero.
How can we check if the word has been guessed and then terminate the loop early? **Hints:**

- We know that the word has not been guessed if dashes are printed for missing letters. Therefore, keep a count of each dash printed in a variable called *missed*.
- If the number of dashes becomes zero then the word has been found and you can terminate the loop. Which keyword can be used to terminate a loop early?

Additional Exercises - Improve the program

5) The player should only lose a turn on a wrong guess. Let them know how many turns are left.
6) Allow the program to work with both lowercase and uppercase letters. Validate the user input.

Challenge Exercises

7) For each wrong guess add part of a **picture** or a letter of a **w**ord (such as 'TIME UP') so that the complete picture or complete word is only displayed if the player runs out of turns.
8) Try different secret words.
9) The program should base the number of turns based on the length of the word.