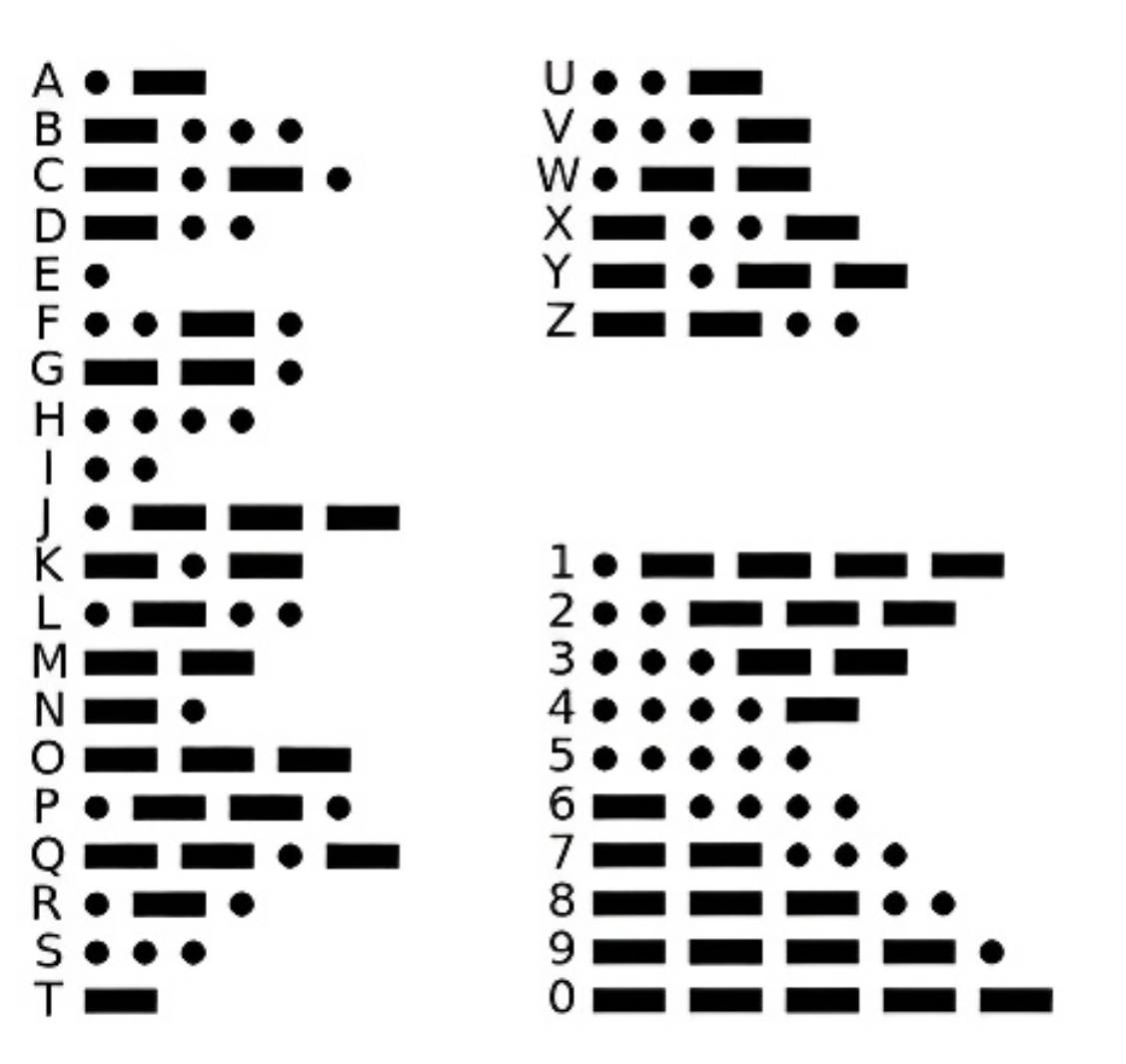
**Morse Code Converter Project Course**

Prerequisites: Python fundamentals, Basic understanding of dictionaries and loops

Version: Python 3

**Introduction**

Morse code is a method used to encode text characters using sequences of dots and dashes to represent letters and numbers. It was widely used in early telecommunication systems and is still employed today in various applications, including amateur radio and signalling.

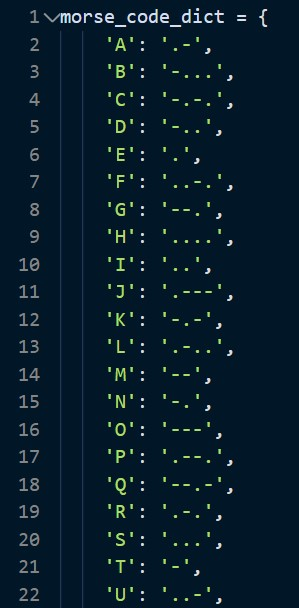
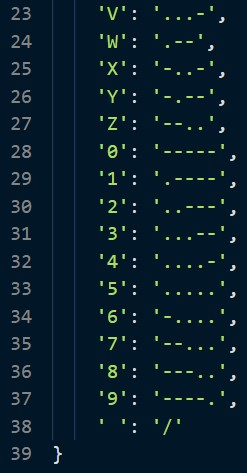


In this tutorial, we will create a Morse Code Converter using Python. We'll develop a program that translates regular text into Morse. Through this project, you'll learn about dictionaries, loops, string manipulation, and basic encoding techniques in Python.

**Setup**

You can use any Python editor for this tutorial. Feel free to write the code in the built-in Python editor or save it to a separate file using an editor of your choice, such as VS Code.

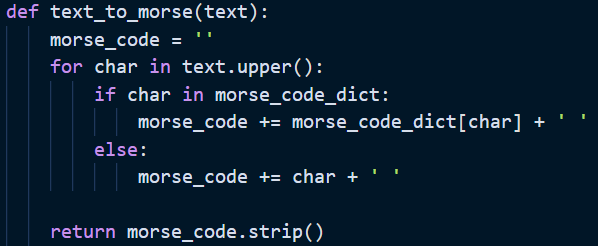
**Step 1: Make a dictionary**

To start, let's create a dictionary called ‘morse\_code\_dict’. This dictionary will contain mappings for   
each letter (A-Z), each digit (0-9), and a space (' ') to their respective Morse code representations.  
Take a moment to review the dictionary and understand how each character is encoded.

**Step 2: Define the function**

In this step, we'll create a function called ‘text\_to\_morse()’ that accepts a text input   
(We will ask this later to the user) and returns its Morse code representation.

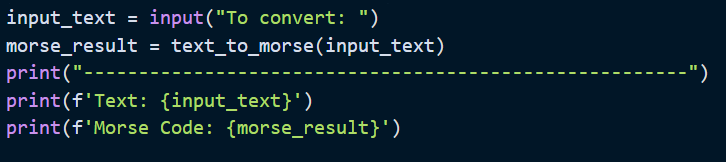


Explanation

* The text\_to\_morse() function takes a text parameter, representing the input text to be converted (We will ask this later to the user).
* We initialize an empty string ‘morse\_code’ to store the Morse code representation of the input text.
* The for loop iterates through each character in the input text after converting it to uppercase using text.upper().
* For each character, we check if it exists in our Morse code dictionary (morse\_code\_dict).
  + If the character is found, we append its Morse code representation followed by a space for the next characters.
  + If the character is not found (e.g., special characters), we append the character itself followed by a space.
* Finally, we use return morse\_code.strip() to remove trailing spaces and return the Morse code representation.

Step 3: Ask input and print!

Let's modify our script to take user input and display the Morse code conversion.



* We use the input() function to prompt the user to enter text for conversion.
* The entered text is stored in the ‘input\_text’ variable.
* We then call our text\_to\_morse() function, passing the user input as an argument, and store the result in the ‘morse\_result’ variable.

Note: In Python, f-strings make it easy to include expressions and variables inside strings. They were introduced in Python 3.6 and provide a simple and readable way to format strings.

Step 4: Run and Test

Save your file and run the code. Test the Morse Code Converter by entering different inputs. After you provide input, it should display the original text and its Morse code representation.

Conclusion

Well done! You've added user input functionality to your Morse Code Converter! Now you can easily convert any text to Morse code using your program.

Bonus: consider adding more features or improving the user interface to make the converter even more user-friendly. Good luck!  
  
[Full solution code](https://github.com/Jasperdj317/Morse/blob/main/morse.py)