Python NATO ICAO Encoder Tutorial

Contents

Python NATO ICAO Encoder Tutorial 1
Requirements 1
Tutorial 1: NATO Dictionary 2
Tutorial 2: NATO Encoder 1 2
Tutorial 3: NATO Encoder OOP 4
pyttsx3 6
Tutorial 4: Text to Speech 6
Tutorial 5: NATO Encoder Text to Speech 7
Extra Extra Credit: GUI 10
Assignment Submission 10

Time required: 60 minutes

- Comment each line of code as shown in the tutorials and other code examples.
- Follow all directions carefully and accurately.
- Think of the directions as minimum requirements.

Requirements

The North America Treaty Organization (NATO) Phonetic Alphabet is the most widely used spelling alphabet. A spelling alphabet (aka radio alphabet, or telephone alphabet) is a set of words used to stand for the letters of an alphabet in oral communication. Each word in the spelling alphabet typically replaces the name of the letter with which it starts. It is used to spell out words when speaking to someone not able to see the speaker, or when the audio channel is not clear.

The International Civil Aviation Organization (ICAO) Alphabet is a series of words which are used to represent each letter of the alphabet. These are used in critical radio communications between airplanes and ground, and between airplanes in flight to avoid misunderstanding.

Tutorial 1: NATO Dictionary

Create a separate folder for this program.

The code words are given in the Python dictionary below. You can copy and paste this dictionary into the Python file.

Create a Python file named: **nato_dictionary.py**

Copy the following code into the file.

```
"""
    filename: nato_dictionary.py
    NATO ICAO phonetic alphabet
"""
encoder_dictionary = {
    'A': 'Alpha',  'B': 'Bravo',   'C': 'Charlie',
    'D': 'Delta',  'E': 'Echo',    'F': 'Foxtrot',
    'G': 'Golf',   'H': 'Hotel',   'I': 'India',
    'J': 'Juliett','K': 'Kilo',    'L': 'Lima',
    'M': 'Mike',   'N': 'November','O': 'Oscar',
    'P': 'Papa',   'Q': 'Quebec',  'R': 'Romeo',
    'S': 'Sierra', 'T': 'Tango',   'U': 'Uniform',
    'V': 'Victor', 'W': 'Whiskey', 'X': 'X-ray',
    'Y': 'Yankee', 'Z': 'Zulu',    '0': 'Zero',
    '1': 'One',    '3': 'Two',     '4': 'Four',
    '5': 'Five',   '6': 'Six',     '7': 'Seven',
    '8': 'Eight',  '9': 'Niner'
}
```

Tutorial 2: NATO Encoder 1

Create a Python file named: **nato_encoder_1.py**

```
1 """
2     Name: nato_encoder_1.py
3     Author:
4     Created:
5     Purpose: Encode words into NATO alphabet
6 """
7 import nato_dictionary
```

Import the NATO dictionary.

```
# Get input from the user
# .upper() converts all letters to upper case for easy
# comparison to the NATO dictionary keys
words = input("Enter a sentence only: ").upper()
```

Get input from the user in a sentence. Convert all characaters into upper case for easy comparison to the NATO dictionary.

```
# Print the original input as upper case
print(words)
```

The string **words** is now all upper case. This keeps the comparison to the dictionary simpler. The program doesn't have to contend with a mixture of upper and lower case letters

```
# Split the sentence into a list of words
# The default split is at the space between the words
# To split by a different character, put it between the " "
words = words.split(" ")
```

Split the sentence into a list of words. Space is the default split in a string. This example explicitly uses a space to show where you would place a different character. If you want to split by another character like a comma, you would use .split(",")

```
# Print the words list to show the split between words
print(words)
```

Print the list before processing for debugging and to show the list of strings after splitting the string by spaces.

```
# Loop through the word list one word at a time
for word in words:

    # Loop through each word in the list one character at a time
    for char in word:

        # Encode the character from the encoder dictionary
        # use char as the key, return the dictionary value
        encoded_char = nato_dictionary.encoder_dictionary.get(char)

        # Print each encoded character
        # end= " " puts a space between each word
        print(encoded_char, end=" ")

    # Print each encoded word on its own line
    print()
```

Process and display the resulting word list.

Example run.

```
Enter a sentence only: This is just fine
['THIS', 'IS', 'JUST', 'FINE']
Tango Hotel India Sierra
India Sierra
Juliett Uniform Sierra Tango
Foxtrot India November Echo
```

Tutorial 3: NATO Encoder OOP

```python
"""
    Name: nato_encoder.py
    Author:
    Created:
    Purpose: OOP Python CLI program to Encode words into NATO alphabet
"""
import nato_dictionary


class NatoEncoder:
    def __init__(self):
        print("+-----------------------------------------+")
        print("|  --   NATO ICAO Alphabet Encoder    --  |")
        print("+-----------------------------------------+")

# ------------------------ NATO ENCODER ----------------------------------#
    def encoder(self):
        """Encode words into the NATO alphabet."""
        # Get input from the user
        # .upper() converts all letters to upper case for easy
        # comparison to the NATO dictionary keys
        words = input("Enter words only: ").upper()

        # Split the sentence into a list of words
        # The default split is at the space between the words
        # To split by a different character, put it between the " "
        words = words.split(" ")

        # Loop through the word list one word at a time
        for word in words:
            # Loop through each word in the list one character at a time
            for char in word:
                # Encode the character from the dictionary
                # using char as the key, returning the dictionary value
                encoded_char = nato_dictionary.encoder_dictionary.get(char)
                # Print each encoded character
                # end= " " replaces \n and puts a space between each word
                print(encoded_char, end=" ")
            # Print each encoded word on its own line
            print()


# Create program object to start program
nato_encoder = NatoEncoder()
# Program menu loop
while True:
    # Call the encoder method
    nato_encoder.encoder()
    menu_choice = input("Encode another (y, n): ")
    if menu_choice == "n":
        break
```

Example run:

```
+----------------------------------------+
|  --   NATO ICAO Alphabet Encoder    --  |
+----------------------------------------+
Enter words only: This is a fun program
Tango Hotel India Sierra
India Sierra
Alpha
Foxtrot Uniform November
Papa Romeo Oscar Golf Romeo Alpha Mike
Encode another (y, n): n
```

pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech.

The pyttsx3 module supports two voices in Windows. One is female and the second is male which is provided by "sapi5" for windows.

There are a lot of possibilities for creative programs using this library.

Pyttxs3 supports three TTS (Text to Speech) engines:

- sapi5 – SAPI5 on Windows
- nsss – NSSpeechSynthesizer on Mac OS X
- espeak – eSpeak on every other platform

Tutorial 4: Text to Speech

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech.

The pyttsx3 module supports two voices in Windows. One is female and the second is male which is provided by "sapi5" for windows.

1. Install the **pyttsx** module: **pip install pyttsx3**

```
"""
    Name: text_to_speech_cli_1.py
    Author:
    Created:
    Purpose: Render text into speech
    This library has many modules with which you can try
    changing the voice, volume, and speed rate of the audio.
    https://pypi.org/project/pyttsx3/
    https://pyttsx3.readthedocs.io/en/latest/
"""
# Linux:   pip3 install pyttsx3
# Windows: pip install pyttsx3
import pyttsx3

# init function creates an engine
# instance/object for speech synthesis
engine = pyttsx3.init()

# Pass text to engine.say method
engine.say("Hello, how may I help you?")

# run and wait method processes the voice
engine.runAndWait()
```

Example run:

In a male voice, your computer should say: "**Hello, how may I help you?**"

Tutorial 5: NATO Encoder Text to Speech

1. Save **nato_encoder.py** as **nato_text_to_speech.py**

The following code modifies the NATO Converter to use text to speech.

```
"""
    Name: nato_text_to_speech.py
    Author:
    Created:
    Purpose: Display and say NATO alphabet encoding
    This library has many modules with which you can try
    changing the voice, volume, and speed rate of the audio.
    https://pypi.org/project/pyttsx3/
    https://pyttsx3.readthedocs.io/en/latest/

"""
# pip install pyttsx3
import pyttsx3
import nato_dictionary


class NatoEncoder:
    def __init__(self):
        # init method creates an engine
        # instance/object for speech synthesis
        self.engine = pyttsx3.init()
        # Constants to change speech engine properties
        RATE = 105    # integer default 200 words per minute
        VOLUME = .7   # float 0.0-1.0 inclusive default 1.0
        VOICE = 1     # Set 1 for Zira (female), 0 for David (male)
        # SET VOICE RATE
        self.engine.setProperty('rate', RATE)
        # SET VOLUME
        self.engine.setProperty('volume', VOLUME)
        # Retrieves all available voices from your system
        voices = self.engine.getProperty('voices')
        # In Windows, set voice to Zira
        self.engine.setProperty('voice', voices[VOICE].id)
```

- Import the pyttsx3 text to speech library and NATO dictionary.
- RATE, VOLUME, and VOICE are 3 of the properties that can be changed. Go to https://pyttsx3.readthedocs.io/en/latest/ for more information.

```
# ---------------------- DISPLAY AND SAY TITLE ----------------------------#
    def title(self):
        """Display and say program title and prompt."""
        print("+-----------------------------------------+")
        print("|     --     NATO Alphabet Encoder     --     |")
        print("+-----------------------------------------+")
        # Pass text to engine.say method
        self.engine.say("NATO Alphabet Encoder")
        self.engine.say("Enter words only")
        # run and wait method processes the voice
        self.engine.runAndWait()
```

```
# ---------------------- GET INPUT ----------------------------------------#
    def input(self):
        """Get input from user."""
        # Get words from user
        self.words = input("Enter words only: ").upper()
```

```
# ---------------------- NATO ENCODER -------------------------------------#
    def encode(self):
        """Encode input to NATO alphabet."""
        # Split the sentence into a list of words
        words = self.words.split()
        encoded_sentence = ""
        # Loop through word list one word at a time
        for word in words:
            print(word.title())
            # Pass text to engine.say method
            self.engine.say(word)
            # run and wait method processes the voice
            self.engine.runAndWait()

            # Loop through each word one char at a time
            for char in word:
                # Encode the character from the dictionary
                # use char as the key, return the dictionary value
                encoded_char = nato_dictionary.encoder_dictionary.get(char)
                # Concatenate encoded_char to encoded_sentence
                # Add a space between each word for
                # speech engine to distinguish words
                encoded_sentence = encoded_sentence + (encoded_char + " ")

            # Print each encoded sentence
            print(encoded_sentence)
            # Pass text to engine.say method
            self.engine.say(encoded_sentence)
            # Clear the sentence string for the next word
            encoded_sentence = ""

        # run and wait method processes the voice
        self.engine.runAndWait()
        # Pass text to engine.say method
        self.engine.say("Encode another?")
        # run and wait method processes the voice
        self.engine.runAndWait()
```

```
# ---------------------- SAY BYE ----------------------------------------#
    def bye(self):
        print("Bye")
        # Pass text to engine.say method
        self.engine.say("Bye.")
        # run and wait method processes the voice
        self.engine.runAndWait()
```

```
# ---------------------- RUN PROGRAM ------------------------------------#
# Create program object to start program
nato_encoder = NatoEncoder()
# Program menu loop
while True:
    nato_encoder.title()
    nato_encoder.input()
    nato_encoder.encode()
    menu_choice = input("Encode another (y, n): ")
    if menu_choice == "n":
        nato_encoder.bye()
        break
```

Example run:

```
+----------------------------------------+
|    --    NATO Alphabet Encoder    --   |
+----------------------------------------+
Enter words only: Java
Java
Juliett Alpha Victor Alpha
Encode another (y, n): n
```

Extra Extra Credit: GUI

Convert this OOP program to a Tkinter GUI.

Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.