Task 2: Building a Chatbot using Python

Problem Statement: Your task involves building a chatbot using Python. You will explore various libraries and techniques to develop a responsive and interactive chatbot that can engage with users effectively.

Code Explanation:
This code implements a simple chatbot using the Natural Language Toolkit (NLTK) library in Python.

- **Importing Required Modules**

```
import nltk as n
from nltk.chat.util import Chat,reflections
```

nltk as n: The `nltk` library is imported with the alias `n`. However, this alias is not used in the rest of the code.
Chat: The `Chat` class is imported from `nltk.chat.util`. This class is used to create the chatbot by passing it a set of patterns and responses.
reflections: The `reflections` dictionary is also imported from `nltk.chat.util`. This dictionary is used to map pronouns and verbs to their corresponding reflections (e.g., "I" to "you", "am" to "are").

- **Defining Conversation Patterns**

```
pairs = [
    ...
]
```

The `pairs` variable is a list of tuples. Each tuple consists of a regular expression pattern and a list of possible responses. The chatbot uses these patterns to match user input and respond accordingly.

Example Patterns and Responses:

Pattern: `r"My name is (.)"`
Responses: `["Hello %1, how are you today?", ...]`
This pattern matches any input starting with "My name is", capturing the name provided by the user. The captured name is inserted into the response where `%1` appears.

Pattern: `r"(hi|hello|hola|holla|hii)(.)"`
Responses: `["Hello", "Hey there"]`
This pattern matches various greetings and responds with a friendly greeting.

- **Defining the `chatbot()` Function**

```
def chatbot():
    print("Hi I am your chatbot. Type 'quit' to exit")
    chat = Chat(pairs, reflections)
    chat.converse()
```

The `chatbot()` function initializes the chatbot and starts a conversation.
chat = Chat(pairs, reflections): The `Chat` object is created using the defined `pairs` and the `reflections` dictionary.
chat.converse(): This method initiates an interactive chat session where the chatbot listens for user input and responds based on the defined patterns.

- **Running the Chatbot**

```
if __name__ == "__main__":
    chatbot()
```

This conditional ensures that the `chatbot()` function is called only if the script is run directly (not imported as a module in another script).

How It Works:

The chatbot will continuously prompt the user for input. It will try to match the input against the patterns defined in `pairs`. If a match is found, the chatbot will select a random response from the list associated with that pattern. The conversation continues until the user types "quit", at which point the chatbot will exit with a farewell message. This is a simple and extensible chatbot framework that can be easily modified by adding more patterns and responses to the `pairs` list. The use of regular expressions allows for flexible input matching, making the chatbot more interactive and responsive to various user inputs.