DISEÑO E IMPLEMENTACIÓN DE CLASES
CHATBOT

HANNA KATHERINE ABRIL GÓNGORA
KAROL ASLEY ORJUELA MAPE

UNIVERSIDAD MANUELA BELTRÁN

PROGRAMACIÓN ORIENTADA A OBJECTOS

DOCENTE

DIANA MARCELA TOQUICA RODRÍGUEZ

BOGOTÁ DC MARTES 27 DE FEBRERO

1. **PREGUNTAS ORIENTADORAS**

   ¿Cuáles fueron los aprendizajes obtenidos al realizar esta guía?, liste como mínimo 3 aprendizajes y relaciónelos con su futuro que hacer profesional.

   - Tener el domino de programar un Chatbot con diferentes preguntas y respuestas. Familiarízate con los conceptos básicos de programación y lenguajes de programación como Python o Java.
   - Aprender a utilizar diferentes técnicas de programación y estructuras para llevar a cabo nuestro Chatbot. Utilizar el lenguaje de programación y la plataforma elegida para implementar el Chatbot y además configurar las interacciones y las respuestas según las preguntas.
   - Tener la capacidad de pensar de forma creativa para diseñar. También teniendo en cuenta si se decidía construir el Chatbot desde cero o inspirarse en alguno existente en plataformas como Stack OverFlow entre otras.
   - También realizar las suficientes pruebas para asegurarse de que funcione correctamente.

2. **EXPLICACIÓN DEL CÓDIGO EN PYTHON**

   - Se importan dos módulos: *random*, que se utiliza para seleccionar respuestas aleatorias, y *re*, que se utiliza para hacer coincidir patrones mediante expresiones regulares. Son de gran importancia ya que gracias a estas se realiza la interacción total del chatbot con el usuario ya que gracias a estos módulos el chatbot es capaz de dar la respuesta adecuada a la pregunta del usuario. Estos de implementan en la función *chatear* de la clase *Chatbot.*

```
1   import random
2   import re
3
```

   - Se define una clase *Preguntas* que almacena preguntas y respuestas en una lista

de listas (es una estructura de datos que contiene otras listas como sus elementos) *llamada _pares*. Cada par de pregunta-respuesta está representado como una lista con la pregunta o expresión clave como un patrón de expresión regular y la respuesta como una lista de posibles respuestas a la pregunta. Estos se establecieron como privados ya que no necesitan ser accedidos directamente desde afuera de la clase. Esta clase con este método es de gran importancia ya que es el corazón del chatbot, es la biblioteca de posibles preguntas que puede hacer el usuario y se genera la respuesta a esa pregunta, esta clase tiene relación directa con la clase *Chatbot.*

```
class Preguntas():
    def __init__(self):
        self._pares = [
            [r"bien?", ["Me alegro que estes bien. Yo estoy feliz de que estes aqui, como te puedo ayudar? "]],

            [r"que es python", ["""Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el ⋯

            [r"comentarios en python", ["""Los comentarios en Python comienzan con el carácter numeral, '#', ⋯

            [r"variables en python", ["""En Python, las variables se definen y se asignan valores utilizando el símbolo ⋯

            [r"tipos de datos de python", ["""En python existen estos tipos de datos: ⋯

            [r"listas en python", ["""Para crear una lista en Python, simplemente hay que encerrar una secuencia ⋯

            [r"if y else en python", ["""En Python, if y else son estructuras de control de flujo que permiten ejecutar ⋯

            [r"bucles for en python", ["""El bucle o ciclo for se utiliza para iterar sobre una objeto ⋯

            [r"bucles while en python", ["""Un bucle while permite repetir la ejecución de un grupo de ⋯

            [r"funciones en python", ["""Para crear una función en Python, puedes usar la palabra reservada def. ⋯

            [r"clases y objetos en python", ["""Para crear una clase vamos a emplear la palabra reservada class seguido⋯

            [r"gracias", ["""De nada, tienes otra pregunta? """]],
        ]
```

- El método *obtener_pares()* devuelve la lista de pares, en donde se encuentra la lista de listas con la relación pregunta-respuesta, esta imprime el resultado a la posible pregunta del usuario.

```
    def obtener_pares(self):
        return self._pares
```

- Se define la clase *Chatbot*.

En el método *__init__*, se instancia un objeto de la clase Preguntas para acceder a los pares de preguntas y respuestas, con los cuales va interactuar el chatbot con el usuario.

```
111   class Chatbot():
112       def __init__(self):
113           self.preguntas = Preguntas()
114
```

- El método *chatear()* simula una conversación entre el chatbot y el usuario.

El bucle *while* True permite al usuario realizar múltiples preguntas hasta que se digite una frase que tenga una coincidencia con "finalizar" en ese momento el chatbot se despedirá y cerrara el programa.

Lo que hace este método es leer la entrada del usuario y busca una coincidencia entre la entrada y los patrones de preguntas definidos anteriormente en la clase *Preguntas*.

Si se encuentra una coincidencia, elige la respuesta que coincide con ese patrón y la retorna o la imprime.

Si no se encuentra una coincidencia, solicita al usuario que reformule su pregunta.

```
115       def chatear(self):
116           print("Hola, ¿cómo estás?")
117           print("")
118           while True:
119               entrada = input("Tú: ")
120               if entrada.lower() == "finalizar":
121                   print("Chatbot: Chao, fue un placer conversar contigo.")
122                   break
123               for patron, respuestas in self.preguntas.obtener_pares():
124                   if re.search(patron, entrada):
125                       print("")
126                       print("Chatbot:", random.choice(respuestas))
127                       print("")
128                       break
129               else:
130                   print("Chatbot: No entiendo, ¿puedes reformular tu pregunta?")
131
```

- Este bloque asegura que el bloque de código debajo solo se ejecute si este script

es ejecutado directamente, no si es importado como un módulo.

Se instancia un objeto de la clase Chatbot y se llama al método *chatear()* para iniciar la conversación del chatbot y por lo tanto el programa.

```
133  if __name__ == "__main__":
134      chatbot = Chatbot()
135      chatbot.chatear()
136
```

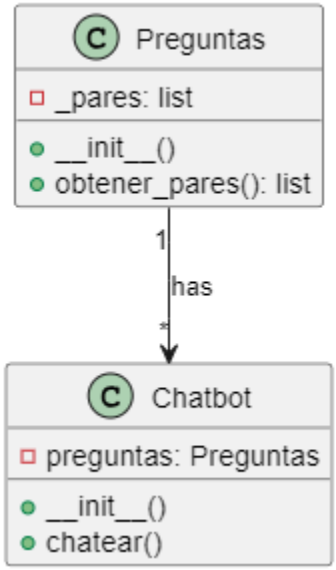
3. **EXPLICACION DEL CODIGO UML**

```
1   @startuml Chatbot
2
3   class Preguntas {
4       - _pares: list
5       + __init__()
6       + obtener_pares(): list
7   }
8
9   class Chatbot {
10      - preguntas: Preguntas
11      + __init__()
12      + chatear()
13  }
14
15  Preguntas "1" ⟶ "*" Chatbot : has
16
17  @enduml
18
```

- Línea 1: Inicia un diagrama UML para un chatbot.
- Línea 2-7: Define una clase llamada “Preguntas”. Esta clase tiene un atributo de lista llamado “_pares” y dos métodos:
    - *init():* Inicializa la clase.
    - *obtener_pares():* Retorna una lista.
- Línea 9-13: Define otra clase llamada *Chatbot* que hereda de la clase “Preguntas”. La clase *Chatbot* tiene los siguientes elementos:
    - Un atributo *preguntas* del tipo *Preguntas*.

    - Un método *init()* para inicializar la clase.
    - Un método *chatear()*.
- Línea 15: Indica una relación entre las clases, donde cada instancia de *Chatbot* tiene múltiples instancias de *Preguntas*.
- Línea 16-18: Finaliza el diagrama UML.

4. DIAGRAMA DEL CODIGO EN UML



5. EXPLICACION Y USO

Este es un proyecto en donde se implementa un pequeño Chatbot, sobre Python, en este caso hay que tener en cuenta que los Chatbots, avanzados maneja IA para su funcionamiento, comprensión de preguntas y obtención de respuestas en una conversación, en este caso, nosotros lo hicimos mediante un método de búsqueda de patrones o palabras clave.

Por ejemplo, al dejar correr el programa y teniendo en cuenta que son preguntas únicamente orientadas sobre el lenguaje de Python, se le pueden preguntar cosas como, que es una variable en Python, o que es un bucle for en Python, y este mediante el análisis del patrón predefinido en el código y la entrada del usuario, te arrojara la respuesta de dicha pregunta.

Hay que tener en cuenta que tiene una librería de preguntas que es pequeña, apenas de 10 preguntas y respuestas de las cuales 9 son de información de Python, hay que destacar que esta se puede expandir con temas diferentes, además el usuario también debe tener en cuenta que se puede hacer una misma preguntas de diferentes maneras pero siempre debe incluir las palabras clave para que el chatbot haga la correspondiente búsqueda de ese patrón, si no se incluyen dichas palabras clave el chatbot te va a informar que formules tu pregunta de otra manera.

6. DEMOSTRACION DE PRUEBAS Y FUNCIONAMIENTO DEL CODIGO

6.1. Conversación 1.

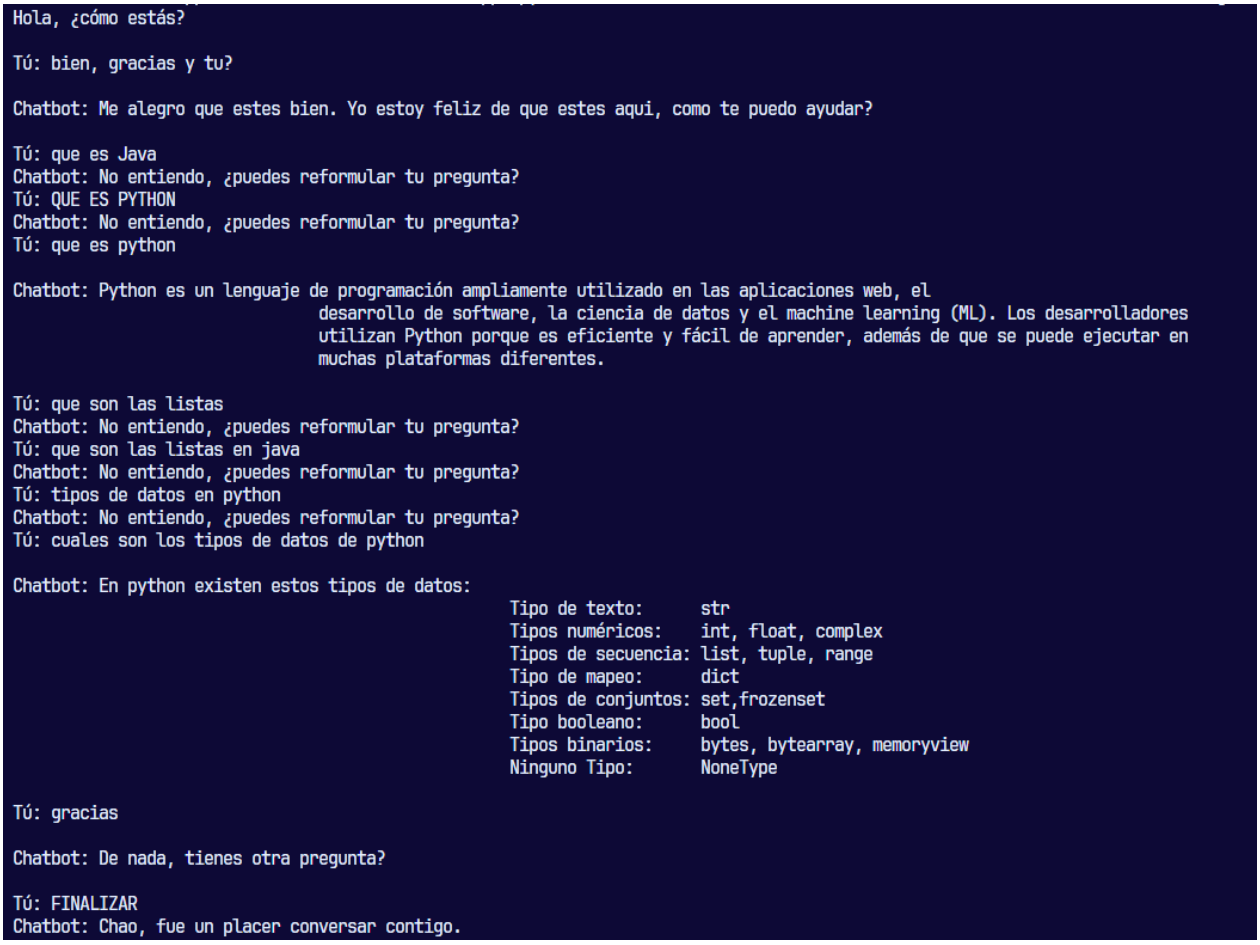
```
Hola, ¿cómo estás?

Tú: muy bien, y tu?

Chatbot: Me alegro que estes bien. Yo estoy feliz de que estes aqui, como te puedo ayudar?

Tú: me puedes decir que es python

Chatbot: Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el
                              desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores
                              utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en
                              muchas plataformas diferentes.

Tú: que es python

Chatbot: Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el
                              desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores
                              utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en
                              muchas plataformas diferentes.

Tú: gracias

Chatbot: De nada, tienes otra pregunta?

Tú: no, finalizar
Chatbot: No entiendo, ¿puedes reformular tu pregunta?
Tú: finalizar
Chatbot: Chao, fue un placer conversar contigo.
```

Se evidencia la interacción usuario-chatbot, en este caso se le realiza una cuestión sobre que es Python, se le realiza la pregunta de dos maneras diferentes, pero con las mismas palabras clave y se evidencia la misma respuesta acertada sobre la pregunta dada.

6.2. Conversación 2.

En esta conversación se presentan otros tipos de preguntas, en donde el chatbot, responde con su respectiva respuesta y ejemplo.

6.3. Conversación 3.



Se ve la siguiente conversación en donde se aprecia que si no se utilizan las palabras clave predefinidas te va a pedir que reformules la pregunta, y también en caso de las preguntas no las detecta si se realiza en mayúsculas (Se olvido tener en cuenta eso al momento de realizar el código) pero si se puede apreciar que al momento de finalizar el chat si detecta las mayúsculas, también se puede apreciar que el código funciona correctamente al momento de realizar una pregunta correcta después de realizar varias incorrectas.

7. DECISIÓN DE DISEÑO O CONSIDERACION IMPORTANTE

Respecto a la decisión del diseño, nos inspiramos de el funcionamiento de un chatbot real por decirlo así, no queríamos que la interacción con el usuario fuera mediante menús con opciones, queríamos que fuera más una conversación así fuera que estuviera limitada por las palabras que usara el usuario, lo cual se logro de manera exitosa. Respecto al tema del chatbot, decidimos que seria bueno un chatbot enfocado a nuestros conocimientos enfocados a la carrera, como lo es el lenguaje de Python, se podría agregar más información sobre otros lenguajes y otras cosas relacionadas a desarrollo de software, pero por el momento tomamos la decisión de centralizar bastante el conocimiento de el chatbot.

Respecto a las consideraciones importantes, llegaron al momento de decidir como haríamos funcionar el chatbot, sabíamos que necesitábamos que el programa buscara coincidencias o patrones en lo que ingresara el usuario para así comparar y dar una respuesta acertada, pero no sabíamos cómo se podía hacer o si siquiera era posible, entonces en nuestra investigación previa encontramos algunos ejemplos de comparación de parejas en Python las cuales se realizaban con listas de listas, y mediante el importe de algunos módulos para cumplir el propósito de búsqueda de coincidencias, lo cual nos fue de una gran utilidad, y mediante a esos ejemplos que encontramos en diversas fuentes supimos como realizar la parte que se podría considerar como más difícil en el proyecto la cual era que el chatbot interactuara de manera correcta con el usuario dependiendo de lo que ingrese este. Lo demás realmente ya fue como la elección de dichas palabras clave que no pueden faltar en una pregunta relacionada a algún tema en específico, y la elección de las respuestas a dichas preguntas, entre otras cosas como hacer más amigable la interacción del usuario y el chatbot, como un saludo, una despedida o preguntas que ayuden a hacer mas continua y amena la interacción.

8. PILARES DE POO IMPLEMENTADOS

En el código, podemos apreciar algunos de los pilares de la Programación Orientada a Objetos (POO) como lo son:

8.1 Clases: En la definición de la clase Preguntas, se utiliza el concepto de clase. Una clase es como un molde que contiene características y acciones con las cuales puedes

construir múltiples objetos. En este caso, Preguntas es una clase que tiene un constructor *(__init__).*

8.2 Propiedades (Atributos): Aunque no se ven explícitamente en el código, las propiedades o
atributos de una clase se definen dentro del constructor *(__init__).* En este caso, *_pares* es un atributo de la clase *Preguntas*.

8.3 Métodos: El método *__init__* es un ejemplo de un método. Los métodos son las acciones que
una clase puede realizar. En tu código, el método *__init__* inicializa el atributo _pares.

8.4 Objetos: Aunque no se crea una instancia de la clase Preguntas en este fragmento de código,
los objetos son instancias de una clase. Por ejemplo, se crea un objeto llamado *preguntas_objeto*, sería una instancia de la clase Preguntas.

9. CONCLUCIONES

Realmente nos gustó mucho el proyecto, nos ayudo a comprender muchos conceptos sobre la programación orientada a objetos como sus pilares principales, explicados anteriormente, no enseño a crear un diseño efectivo en donde una interacción fluida pudiera ser posible, también aprendimos conceptos nuevos como los módulos que se importaron su importancia y para que servían además de el aprendizaje del manejo de lista de listas que se había manejado en programación básica, pero que se le vio un enfoque importante y muy útil en esta situación. Nos emociono mucho la idea de haber podido crear un “mini chat GPT” por decirlo así, ya que emula mas o menos el funcionamiento de Chatbots reales.