HELLENIC
REPUBLIC
UNIVERSITY
OF MACEDONIA

Project: First Aid Chatbot

Kadrefi Kiriaki- Maria

This project was implemented as part of the Computation Theory course of the Department of Applied Informatics.

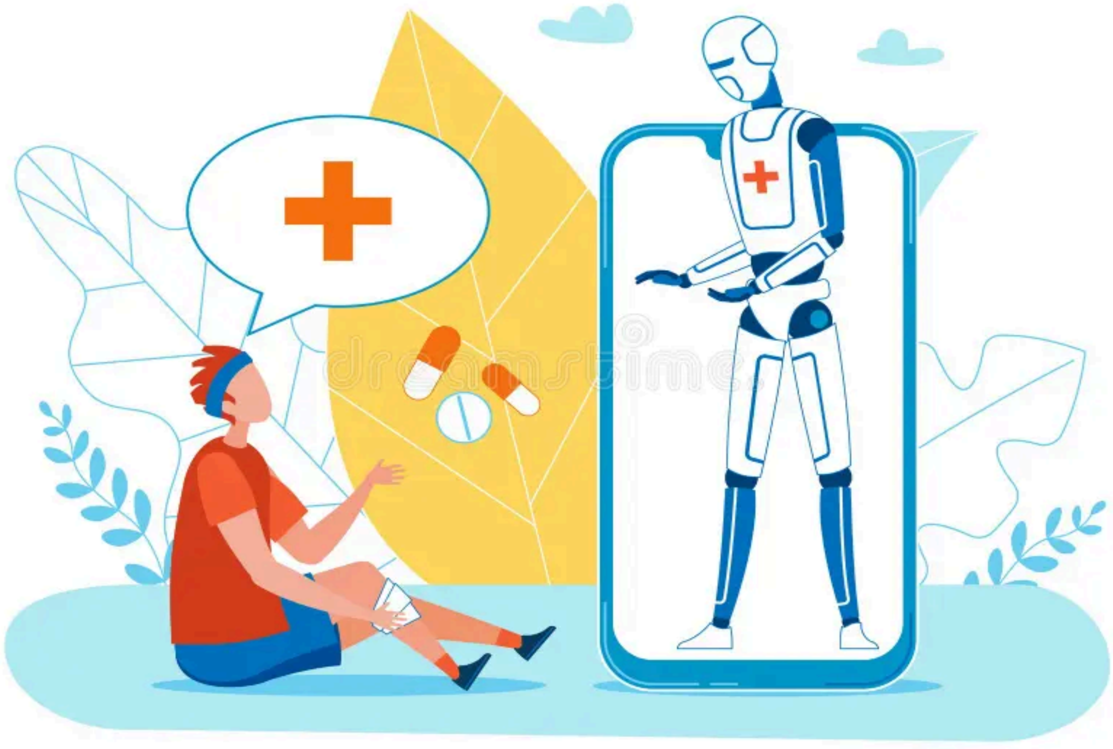
Instructor: Ioannis Refanidis

Table of Contents

1. Development of a First Aid Chatbot for Emergency Guidance 3
2. Functionality Overview 4
3. Code Explanation 5
    Code Execution 5
    Explanation of Regular Expressions 6
    Significance of Regular Expression Priority in Chatbot Response Accuracy 8

1. Development of a First Aid Chatbot for Emergency Guidance

The topic chosen for the development of the chatbot concerns First Aid, a critically important field that can save lives and provide immediate relief in emergency situations. A chatbot in this domain can offer quick guidance to users by providing advice and instructions on how to handle accidents or illness, even in dangerous situations. The programming language chosen for implementation is Python, due to its ease of use and the libraries it offers. One of these is **re** (Regular Expressions), which allows the detection and analysis of specific text patterns. Thus, the chatbot is capable of recognizing emergency situations and responding with appropriate advice. In this way, it can help users react correctly and efficiently in dangerous situations.



2. Functionality Overview

This chatbot supports the understanding of predefined sentence patterns, with the goal of recognizing important information from the user while allowing for some variation in phrasing. If a sentence is not understood, an appropriate message is displayed.

Some example sentences/questions include:

- **Hi, I need your help!**
- **Someone passed out! What should I do?**
- **What is the way to perform CPR?**
- **What is the emergency number?**
- **What to do if someone is bleeding?**
- **What should I do during a seizure?**
- **How many compressions should I give during CPR?**

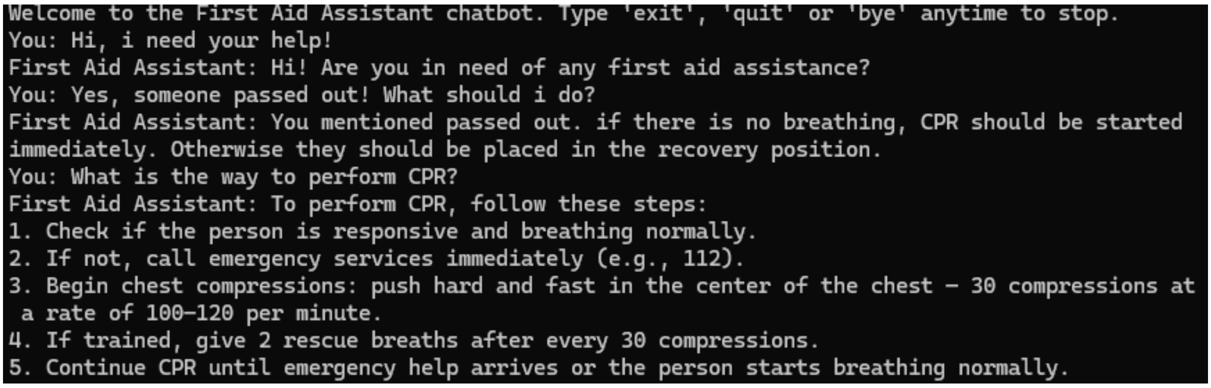
After the user types the desired message, it is matched with a predefined language pattern (pattern) using the **re** library. For implementing the logic of understanding the user's inputs, regular expressions are used, allowing flexible recognition of patterns within the text. Once a match is found, a random response from the pre-saved answers related to that specific pattern is selected. The interaction can continue normally until the user explicitly states their desire to end the conversation (by typing "exit", "quit", or "bye").

3. Code Explanation

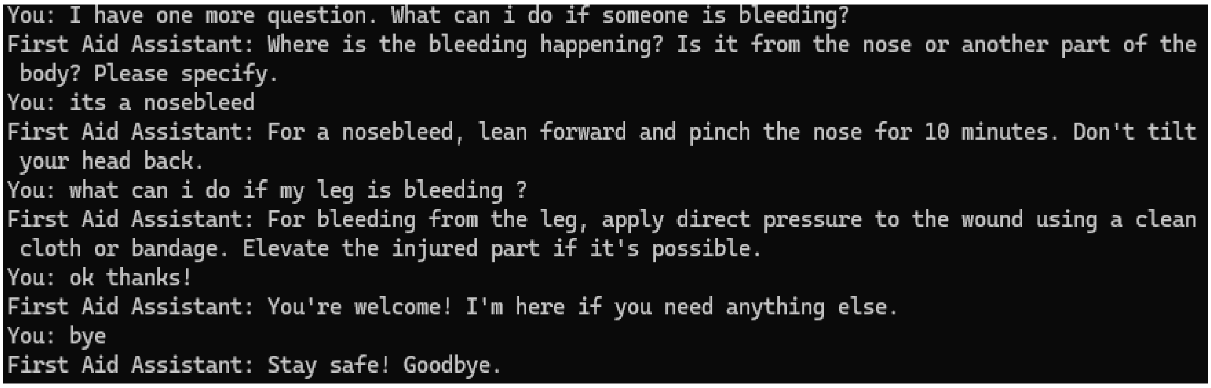
Code Execution

To run the code, execute the command **python first_aid_chatbot.py** in the terminal. Below are some screenshots from the code execution with specific comments:

An example of a complete conversation with the chatbot:

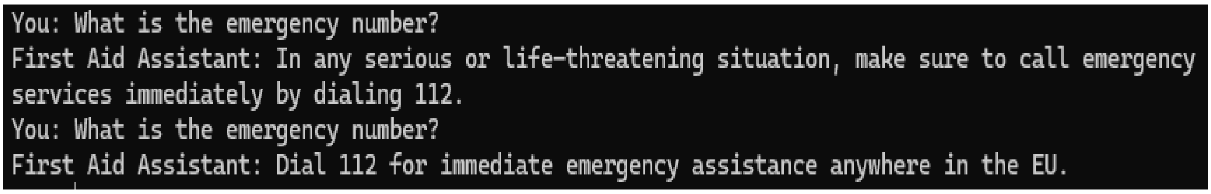


```
Welcome to the First Aid Assistant chatbot. Type 'exit', 'quit' or 'bye' anytime to stop.
You: Hi, i need your help!
First Aid Assistant: Hi! Are you in need of any first aid assistance?
You: Yes, someone passed out! What should i do?
First Aid Assistant: You mentioned passed out. if there is no breathing, CPR should be started
immediately. Otherwise they should be placed in the recovery position.
You: What is the way to perform CPR?
First Aid Assistant: To perform CPR, follow these steps:
1. Check if the person is responsive and breathing normally.
2. If not, call emergency services immediately (e.g., 112).
3. Begin chest compressions: push hard and fast in the center of the chest - 30 compressions at
 a rate of 100-120 per minute.
4. If trained, give 2 rescue breaths after every 30 compressions.
5. Continue CPR until emergency help arrives or the person starts breathing normally.
```



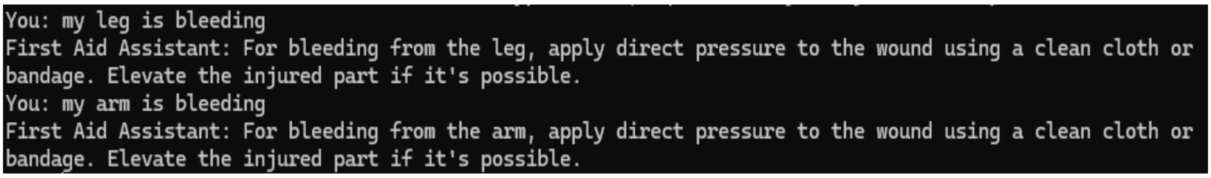
```
You: I have one more question. What can i do if someone is bleeding?
First Aid Assistant: Where is the bleeding happening? Is it from the nose or another part of the
 body? Please specify.
You: its a nosebleed
First Aid Assistant: For a nosebleed, lean forward and pinch the nose for 10 minutes. Don't tilt
 your head back.
You: what can i do if my leg is bleeding ?
First Aid Assistant: For bleeding from the leg, apply direct pressure to the wound using a clean
 cloth or bandage. Elevate the injured part if it's possible.
You: ok thanks!
First Aid Assistant: You're welcome! I'm here if you need anything else.
You: bye
First Aid Assistant: Stay safe! Goodbye.
```

Displaying Different Responses for Repeated Inputs:



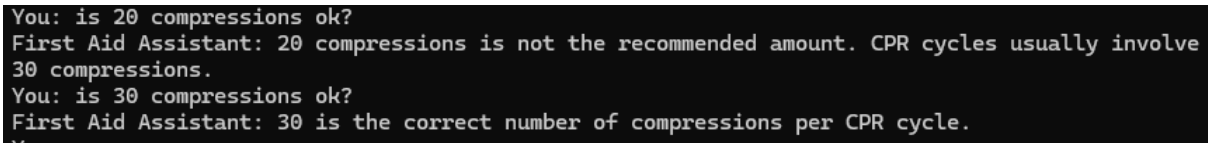
```
You: What is the emergency number?
First Aid Assistant: In any serious or life-threatening situation, make sure to call emergency
services immediately by dialing 112.
You: What is the emergency number?
First Aid Assistant: Dial 112 for immediate emergency assistance anywhere in the EU.
```

Modifying Responses Based on Injury Location:



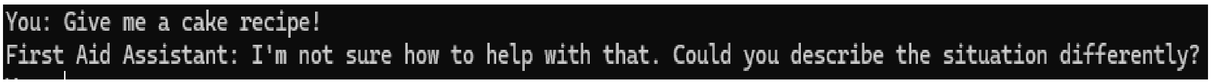
```
You: my leg is bleeding
First Aid Assistant: For bleeding from the leg, apply direct pressure to the wound using a clean cloth or
bandage. Elevate the injured part if it's possible.
You: my arm is bleeding
First Aid Assistant: For bleeding from the arm, apply direct pressure to the wound using a clean cloth or
bandage. Elevate the injured part if it's possible.
```

Recognition of the Correct Compression Count in CPR:



```
You: is 20 compressions ok?
First Aid Assistant: 20 compressions is not the recommended amount. CPR cycles usually involve
30 compressions.
You: is 30 compressions ok?
First Aid Assistant: 30 is the correct number of compressions per CPR cycle.
```

Handling Unrecognized User Input:



```
You: Give me a cake recipe!
First Aid Assistant: I'm not sure how to help with that. Could you describe the situation differently?
```

Explanation of Regular Expressions

Below is the analysis of the regular expressions used in the code development.

1. **(r’\b(hello|hey|hi)\b', [...])**

Explanation:

- This regular expression recognizes any of the following words: **"hello", "hey", ή "hi".**

- **|** : Acts as the "OR" operator in regular expressions and means that the pattern can match any of these words (hello, hey, hi)).

- **word boundary \b :** Allows the chatbot to recognize the word "hi" or other greetings only when they are standalone words, e.g., "hi" and not as part of other text, such as "hwkashwdahiajkfa".

2. **(r'.*\b(pass(ing|ed)? out|los(ing|t)?consciousness|faint(ing|ed)?\b)(.*)', [...])**

Explanation:

- **.*** : in the regular expression means it can match any number of characters, including zero characters (i.e., it can also match if there are no characters at that position).

- The syntax of **pass(ing|ed)?** out and **los(ing|t)?** consciousness are examples of using regular expressions with symbols to detect different variations of words or phrases. For example, **(ing|ed):** This group is inside parentheses and uses **|** to allow the choice between two words (passing, passed). Therefore, (ing|ed) allows the recognition of the forms "pass", "passing", and "passed".

- **?**: This symbol means that the preceding part of the expression (i.e., (ing|ed)) is **optional.**

3. **r'\D*\b(30|how many|number of)\b\D*(compressions|CPR)\D*', ["{0} ..."]**

Explanation:

- It recognizes phrases that contain **exactly the number 30 or questions about the appropriate number of compressions**, in combination with the words "compressions" or "CPR"

- **\D*** : Recognizes any character (zero or more times) except for numbers.

4. **(r'\D*\b(([0-2]?[0-9]|[3-9][1-9]))\b\D*(compressions|CPR)\D*', ["{0} ..."])**

Explanation:

- It detects all "incorrect" CPR numbers (those that differ from 30) so that a corrective response can be triggered

- **(([0-2]?[0-9]|[3-9][1-9]))** : It is used to match integer numbers from 0 to 29 and from 31 to 99, excluding the number 30. The part **[0-2]?[0-9]** matches single-digit numbers (0-9), two-digit numbers from 10-29, and the range 00-09 because the first digit (0-2) is optional (**?**). The part **[3-9][1-9]** matches numbers from 31 to 99, excluding 30 (because there is no 0 in the second position).

5. **An indicative example (r'.*\b(seizure|epilepsy)\b.*', [...])**

Explanation:

- This regular expression will match any text that contains the word "seizure" or "epilepsy", regardless of what other content appears before or after these words.

Significance of Regular Expression Priority in Chatbot Response Accuracy

In the file **first_aid_chatbot.py**, the order of the regular expressions plays a crucial role in ensuring the accuracy of the chatbot’s responses, as the matching process is executed sequentially—from the first to the last expression encountered in the code. This is particularly evident in two areas: the recognition of bleeding incidents and the validation of the correct number of chest compressions during CPR.

In the case of bleeding, the expressions that detect nosebleeds or bleeding at specific body parts come first, followed by a general reference to the word "bleeding". As a result, if the user reports, for example, bleeding on the leg, the chatbot will immediately provide the appropriate advice without requesting further clarification. If the order were reversed, the chatbot would respond with a general follow-up question, even if it already had the necessary information.

Similarly, regarding the number of chest compressions for CPR, the code first checks if the user refers to the correct number—30 compressions—either through a direct question or a simple mention. Only afterwards does it check for incorrect values. This approach ensures that questions such as "What’s the correct number of compressions in CPR?" will not receive a negative or misleading response due to incorrect expression prioritization.