```
    try:
        speech_text = r.recognize_google(audio).lower().replace("'", "")
        print("Melissa thinks you said '" + speech_text + "'")
    except sr.UnknownValueError:
        print("Melissa could not understand audio")
    except sr.RequestError as e:
        print("Could not request results from Google Speech Recognition
        service; {0}".format(e))

    brain(name, speech_text)

main()
```

First, you import the `brain()` function from `brain.py`. Inside the `main()` function, you add `brain()`, where you pass the `name` and `speech_text` arguments.

Your program is now ready to run and to be tested. Go to the terminal, and start the program by issuing the following command:

```
$ python main.py
```

When you see the "Say something!" message, say "Who are you?" into the microphone. You should get the following reply: "I am Melissa, your lovely personal assistant." Try saying something else, and you should receive the following reply: "I dont know what that means!"

There are two problems with the existing system:

1. The library of conversation clauses is limited and static.
2. The recognition system in the brain is very poor, because it compares strings.

You can solve the first problem by having an array of messages and using the `random.choice()` function to answer the user's question. The second problem is much more complex in nature. Even if the user says something like, "Hey, who are you?" the logical engine will pass control to the `undefined()` function. This shouldn't be the case, because "Who are you?" and "Hey, who are you?" essentially mean the same thing. This problem can be handled by checking `speech_text` for certain keywords.

# Fixing Limitation 1

Let's edit `general_conversations.py` to implement the fix just discussed and include some new conversation snippets:

```
import random
from SenseCells.tts import tts

def who_are_you():
    messages = ['I am Melissa, your lovely personal assistant.',
    'Melissa, didnt I tell you before?',
```