**Lesson Overview**

This lesson plan provides a general outline and tips to teach the python Chatbot.

|  |
| --- |
| **Lesson Summary**  **DURATION: 45-60 mins**  **Getting Started: (2-5 mins)**   * Introduce the activity * Direct student to the activity   **Activity: (20-40 mins)**   * Facilitate and support students to modify/interact with the chatbot   **Wrap-up: (5-10 mins)**   * Debrief and close   **Assessment/Extended Learning: (2-5 mins)**   * Optional |

**Audience**

This lesson plan is intended for use with learners of any age who are interested in computer science.

**Learning Objectives**

By participating in this lesson, participants will:

* Python program layout
* Python program input and output (focus on this objective for younger and less accomplished learners and stick with interaction as a way to discuss machine learning)
* Python file loading
* Python chatterbot library usage (a differentiating topic for the faster more accomplished in coding students)
* Python control and decision logic

**Facilitation Guide**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Materials, Resources and Preparation**

* Load the latest Python distribution on the students computers <https://www.python.org/downloads/> provides download options for Windows, Linux, and Mac. Be careful to look for the option to setup
* Download DrSeussBot.py and AllSeuss.txt from: <https://github.com/gregbmeyer/Chatbotwithfeedback>
* Install the chatterbot library for Python (for windows this is done from the command line (C prompt)) additional guidance is here: <https://docs.python.org/3/installing/index.html>

*python -m pip install chatterbot*

**Getting Started (2-5 mins)**

**Introduce the activity (2-5 minutes)**

Have students indicate what kind of chatbots they have encountered. Did they notice limitations in them? Discuss Amazon Alexa, Siri, Google Assistant/Home, GPS driving directions. You can utilize this overview of some innovative chatbots to spark discussion: <https://www.wordstream.com/blog/ws/2017/10/04/chatbots>

**Have students load the code on their laptops/desktops from a student accessible share drive/cloud drive (1 minute)**

* Once downloaded locally, have them open the program and discuss the programming features (library reference, input file load, if/then, while/loop etc, output text). If the chatbot interaction is within the context of a computer programming class then challenge them to assess line by line what is going on in the program, otherwise, just talk about it from a logical higher perspective of here is where we get input and here is where the chatbot decides what to say back and here is where it loops to listen to your next input.
* Outline where the additional do it yourself corpus is loaded. This is one thing that could easily be changed by the students if they want to focus on the conversational learning input to the program. They would open the existing AllSeuss.txt file and modify it or create their own .txt file following the same conversation pattern inside it.
* Example:

**Hello how are you**

**Hi I am fine**

**Where are you from**

**I am from San Antonio**

**Activity (20-40 mins)**

**Facilitate and support students to complete the tutorial, alone or in groups**

**When your students come across difficulties**

It’s okay to respond:

* “I don’t know. Let’s figure this out together.”
* “Technology doesn’t always work out the way we want.”
* “Learning to program is like learning a new language; you won’t be fluent right away.”

**What to do if a student finishes early?**

* Have them explore the other functions available in the chatterbot github project online and implement one or more of them. <https://github.com/gunthercox/ChatterBot>

**Wrap-Up (5-10 mins)**

**Debrief & Close**

* Debrief the activity.
* Share photos and videos of your chatbot session on social media.

**Other ideas**

* Do a gallery walk so students can interact with each other’s work.
* Do a “Think-Pair-Share” to allow students to reflect individually, discuss with a partner and share out as a group.

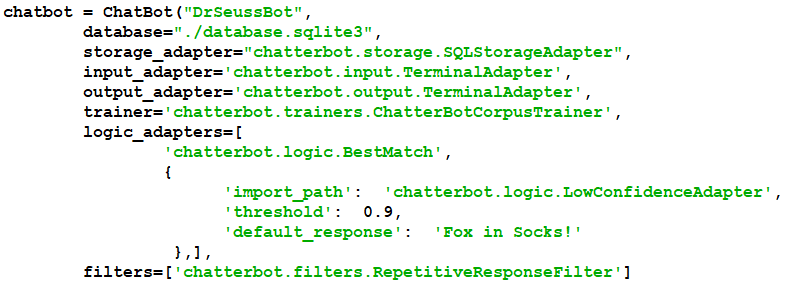
**Assessment/Extended Learning (2-5 mins)**

**Optional**

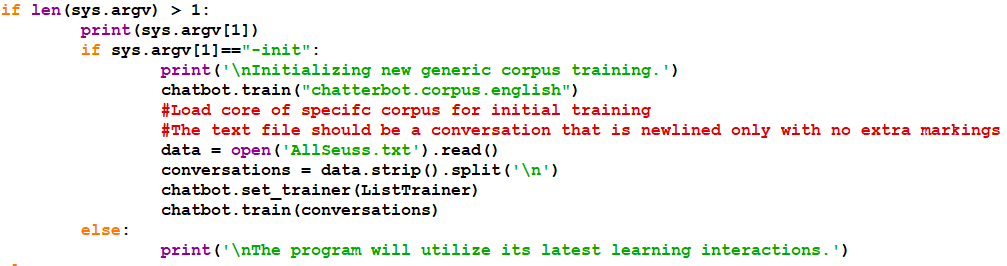
Time permitting, challenge your students to reflect on the day’s activities and continue their learning. Consider:

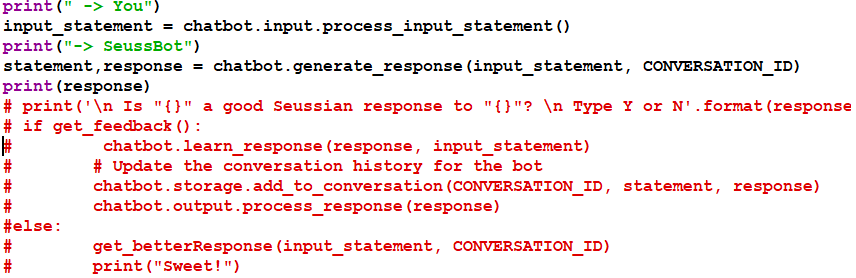
* **Exit Ticket.** Have students complete an exit ticket (list the code control/decision types used in the chatbot python code) before leaving to assess learning.
* **Flip your classroom**. Challenge students to explore the other functions available in the chatterbot github project online and implement one or more of them at home.
* **Writing prompt.** Have students journal at home about what they learned and what they would like to change about the chatbot functions or how a chatbot could help them or their family in some way.

The following text is a set of instructions regarding various pieces of Python code in the program:

  
Most of the parameters shown here should be left alone. The first one simply names your Chatbot. You can change it to any name you prefer. The second one specifies the location and name of the database that is automatically created when you first run this program. It will be located in the same folder as the program itself assuming your current directory setting points to that folder. I recommend leaving it there. The database is preserved between runs. Any updates you enter are preserved in the database. If, at some point, you wish to start over, simply delete the database file. A new one will automatically be created during the next run.

Near the bottom is a default response for entries the Chatbot cannot answer. You may want to change it to something more appropriate to your dialog. Just above the default response is a threshold number. It is currently set at 0.9. The maximum is 1.0. You might try reducing this number. The default is 0.65. A lower number will be more forgiving and will be less likely to use the default response. Of course, the lower this number the more likely you are to get an incorrect answer. The remaining parameters are a bit more complex and are not necessary to change. If you would like to experiment, read the documentation at <https://chatterbot.readthedocs.io/en/stable> before doing so.

  
The first run of the program should specify -init as a command-line parameter. This will cause the program to read your text file and process it into the database which holds all the conversation material. Change the open statement to specify the name and, if necessary, the location of your text file. If the file is located in the same folder as this program, you don't need to specify a location. It is important that your text file is created with a text editor such as Windows Notepad and not a word processor.

  
This section of the program gets you to exercise your conversation. If the Chatbot can find a response very close to your entry, it will supply that response. Otherwise, it will supply the default response. If you would like to supply a more appropriate response for it to use, you can uncomment the statements shown here. This code allows you to update the database with a more appropriate response. That way you don't have to rerun the entire text file to get a few changes made.