**CS 171, Winter, 2021  
Artificial Intelligence  
Project, part 4**

This is part 4 of the quarter-long chatbot project. It is due at the end of the day, 11:59pm Pacific Time, on Monday, March 15. Upload to Canvas, in the "Project Part 4" assignment, a single zip file. The zip file should contain a report file, named Report.txt or Report.docx or Report.pdf, and all your source code files.

You must work on the project individually. You can talk with other people in the class about your and their general approaches, but do not copy any specific ideas or text.

Part 4 of the project is scored on a 25 point scale. Most of the score will be based on the scope and quality of your chatbot implementation. Features of a strong implementation include:

1. All code runs without compiler or runtime errors.
2. All computer code is written by the student or is taken from Part 1 of the project. APIs can be used to access information but not to do natural language processing.
3. The chatbot can accept questions or statements in conversational English. The chatbot can accept a variety of grammatical forms.
4. The chatbot responds in straightforward English sentences. Not all responses need to be full sentences; natural sounding single word or sentence fragment replies can be completely satisfactory.
5. The chatbot can hold a conversation, meaning that it can (to a very limited degree, no doubt) remember the topic under discussion and respond accordingly.
6. The chatbot responds in a friendly, helpful way when the input from the user contains unrecognized words, unparsable grammar, unexpected characters, or any other impediment to a regular response. (It's fine to limit the scope of the expected input; for instance, to accept only lower case characters, or to convert all input to lower case.)
7. The chatbot has helpful and interesting information to provide.
8. The scope of the chatbot's capabilities follows reasonably closely to the goals speficified in parts 2 and 3 of the project.

Some of your Part 4 score will be based on the quality of your report. Carefully follow the directions below.

Your report file for Part 4 should be based on your report file for Part 3, with additional text showing all modifications compared to what you are actually turning in for Part 4. Follow the same structure as in the Part 3 report, and include all the same text (which should have included the Part 2 text). At the end of each section, add either "No changes", or notes on small changes you have made, or if necessary a completely revised version. In step 2, "Technical Overview", include, if appropriate, instructions for how to download and install any additional software necessary to run your implementation. For instance, you might include in your zip a [requirements.txt](https://pip.pypa.io/en/stable/reference/pip_install/#requirements-file-format) file, in which case you should note that in your step 2 write-up. Pay particular attention to your update of step 3, "Phases" -- this should provide a candid and accurate account of your original plans and actual accomplishments. The list of sections below includes a sample "Part 4 update" for each section (plus the "Part 3 updates" that were, in this example, included in the previous part).

1. **Introduction.** Provide a one or two paragraph overview of your chatbot project. What parts of the world will it chat about? What motivated you to select this particular domain? Are there any interesting features you want to highlight?  
   Part 3 update: I've decided to omit the "thunderstorm background sound" from my chatbot when it forecasts stormy weather.  
   **Part 4 update:** No changes.
2. **Technical Overview.** What programming language(s) will you use? Will you use APIs? You should not use external libraries; if you have any uncertainty about whether something written by someone else is acceptable, contact Prof. Frost in advance. Further clarification: It is fine to use an API to access data that your chatbot will chat about. For instance, Google Maps has an extensive API that can be used to answer questions about the distance between two locations; you could use this in your project. But don't use an API or library that is designed specifically for a chatbot.  
   Part 3 update: No changes.  
   **Part 4 update:** I used the API that Prof. Frost approved. This turned out to be much trickier than I expected, so the chatbot is only using it for the thunderstorm warning feature, and all other data is hard-coded into a Python dictionary. Please use the requirements.txt in the zip to download the API software.
3. **Phases.** Divide your implementation (which will comprise parts 3 and 4 of the project) into between 2 and 4 programming phases. Each phase should be a complete and working subset of your chatbot, building on the code from previous phases (and from part 1 of the project, if you so choose). Indicate which phase(s) you will implement for Part 3 (tentatively due March 1) and which remaining phases you will implement for Part 4 (tentatively due March 13). Do not include "blue sky" phases that you do not plan to complete this quarter.  
   Part 3 update: I'm a little behind schedule. All of phase 1 is implemented and working, but in phase 2 I've decided to leave out thunder sounds, as mentioned above, and I haven't gotten to mudslide prediction feature, but that is still in the plans for the completed chatbot. I still plan to have all of phase 3 and phase 4 in the final chatbot.  
   **Part 4 update:**All of phases 3 and 4 are implemented, except that the chatbot only knows about the 10 largest cities in California (hard coded in), and does not use the API to look up information about other cities.
4. **Examples.** Write down examples of interactions between a human and your chatbot. Can questions or answers in the conversation impact subsequent answers by the chatbot? If so, include examples. You are free to make substantial changes later in the project, but these samples should convey a clear sense of your plans for the chatbot.  
   Part 3 update: No changes.  
   **Part 4 update:** No changes.
5. **Input Handling.** Your project will no doubt include code to convert text inputs from the user into some kind of internal representation. Outline the design of this code. Creating a parse tree is optional. If you plan on using a grammar, provide a partial version here. If you plan on following an algorithm described in the textbook or from another source, specify your sources. If you are inventing a novel approach, state that and give an overview.  
   Part 3 update: I've completely rethought my input handling, so everything written above is out of date. Instead, I'm using my CYKParse from earlier in the project, with additional features to recognize the pluperfect subjunctive verb tense.  
   **Part 4 update:** Implemented as planned in Part 3.
6. **Internal Representations and Data Sources.** Outline your plans to represent and process data internally. Options might include (some combination of) logic, data structures, and tables. Will all information used by the chatbot be hard-coded in your code? If your chatbot will access an internal or network-available database in real time, describe that. Describe any interesting capabilities of your chatbot.  
   Part 3 update: Accessing the API turned out to require much more time than expected, but I've got it working and everything above still is accurate.  
   **Part 4 update:** As mentioned above, added an extra Python dict and some hard-coded city names to make up for reduced API usage.
7. **Output handling.** Outline the design of your code that generates English language replies to the user. See "Input Handling" for things to write about.  
   Part 3 update: No changes.  
   **Part 4 update:** No changes.

At the end of your report, include one or more screenshots showing your chatbot running, with inputs (from you the user) and outputs (from the chatbot). Make sure that you include enough conversation for the grader to assess items A - H above.