ENGR 212 Programming Practice
Fall 2016
Mini Project 1
(130 points)

October 10, 2016

In this mini project, you are going to develop a program, given an input text file, use a Markov Chain algorithm to restructure the text, and save the results to another text file. Details regarding the requirements are as follows:

1. Your program will have a graphical user interface (GUI) which will look like as shown in Figure 1. You may use any coloring-scheme you like. The below one is for illustration purposes only. You can use any proper layout manager to arrange your widgets on the screen, though using the place geometry manager is strongly discouraged.

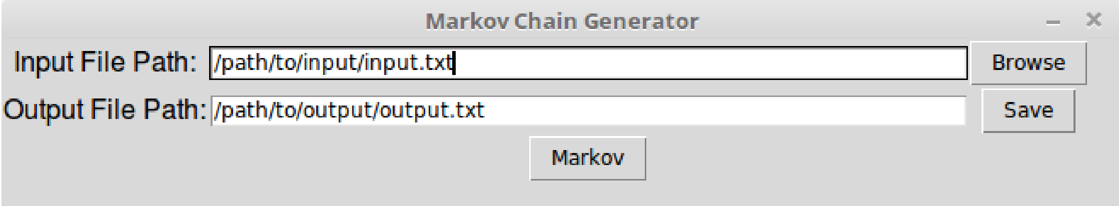
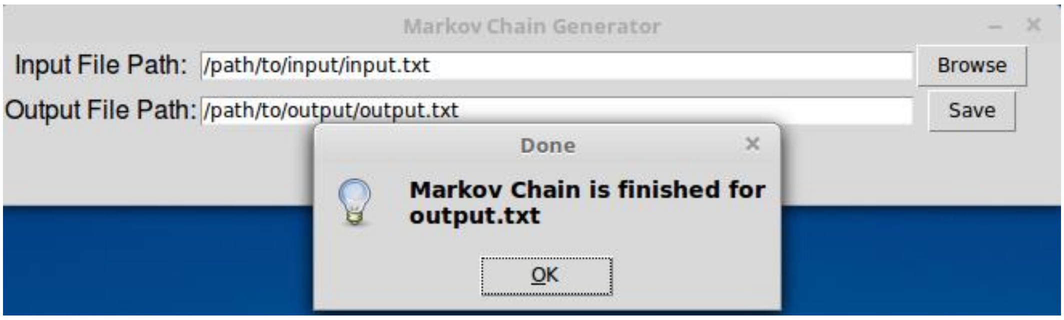


Figure 1

Details of the working of your program are given below:

- Your program **shall** accept the text file to be processed, and the name of the text file to be generated which will hold the text generated by the Markov Chain process as input.
- Then, the user will press the "Markov" button to initiate the conversion process. As a result, the output file **should be** generated by your program.
- When the Markov Chain process is finished, and the output file has been produced, your program **should inform** the user about this by displaying the message as shown below in Figure 2.

2. The user **should be** able to select another file, and then click on the "Markov" button again to process another input file and generate another file as output. Note that, the message indicating the end of the Markov Chain process **should** disappear at the start of each Markov generation, and reappear whenever the output file has been produced correctly.



Markov Chain is finished for output.txt

Figure 2

What is a Markov Chain? :

- A Markov Chain is a way to predict "what to do next" based on "what you did recently" (the details of the process can be found on http://en.wikipedia.org/wiki/Markov_chain). A Markov Chain maintains a fixed series of states, and based on those states determines what to do next, something like walking through a path. The number of steps that are remembered is fixed at the beginning of the process. Such a method can easily be applied to bodies of text to create, humorous constructions of reasonably sensible sentences. This construction is based on the sampling of existing text to create a Markov chain. An existing document is scanned, observing the previous ***n*** words, and remembering the word that follows. Using this information, one can create a new document based on the Markov chain created from the previous document.

  In this project, you are asked to construct a Markov chain that represents a Markov chain of a document given as input to your program. For the purpose of the assignment you can set the value of ***n*** to two. Thus, for every word in the given document, you should record and keep in an appropriate data structure, the previous two words in the document (Please note that, same two word combination may occur before several different words in the same document; hence your data structure should be able to handle a list of words corresponding to the same word pairs).

  To create a new text document, the process begins by selecting the first two words from the original text file as the first two words of the new document. The word recorded originally corresponding to this two word combination will be the third word of the new document. If there are more than one words corresponding to the two word combination, then one of them is selected as randomly. Then, the second and third words from the new document are selected to form a pair, and the word recorded originally corresponding to this new two word combination is added as the fourth word to the new document. The steps are illustrated in an abstract form below:

  ```
  Word1 Word2 markovChain[Word1,Word2] markovChain[Word2,markovChain[Word1,Word2]]…
  ```

  This process continues until a selected two word combination does not have any corresponding words recorded originally. In order to prevent the process to enter an infinite loop, the new document formed can be limited to 500 words, for example. Finally, the generated document should be stored in another text file, as the output of the program.

- Before starting any coding, decide on what Python data structure to use in order to store your Markov chain. State your decision explicitly on your solution (in the form of comment lines). You need not worry about the correctness of the punctuation characters output to the new document. You can test your program by using the example input text files given in the LMS folder for the assignment. Test your program also with other text files of your own.

Can you provide any further pointers that may be helpful? :

- Our coverage in the class did not include using a file dialog (tkFileDialog), but it is really simple, and the following link contains many useful examples of using tkFileDialog.

  http://tkinter.unpythonic.net/wiki/tkFileDialog

- See the following link on how to create a simple message dialog box:

  http://www.tutorialspoint.com/python/tk_messagebox.htm

Warnings:

- **Do not** talk to your classmates on project topics when you are implementing your projects. **Do not** show or email your code to others. If you need help, talk to your TAs or myself, not to your classmates. If somebody asks you for help, explain them the lecture slides, but do not explain any project related topic or solution. Any similarity in your source codes will have **serious** consequences for both parties.
- Carefully read the project document, and pay special attention to sentences that involve "**should**", "**should not**", "**do not**", and other underlined/bold font statements.
- If you use code from a resource (web site, book, etc.), make sure that you reference those resource at the top of your source code file in the form of comments. You should give details of which part of your code is from what resource. Failing to do so **may result in** plagiarism investigation.
- Even if you work as a group of two students, each member of the team should know every line of the code well. Hence, it is **important** to understand all the details in your submitted code. You may be interviewed about any part of your code.

How and when do I submit my project? :

- Projects may be done individually or as a small group of two students (doing it individually is recommended). If you are doing it as a group, only **one** of the members should submit the project. File name will tell us group members (Please see the next item for details).
- Submit your own code in a **single** Python file. Name it with your and your partner's first and last names (see below for naming).
  - If your team members are Deniz Barış and Ahmet Çalışkan, then name your code file as deniz_baris_ahmet_caliskan.py (Do **not** use any Turkish characters in file name).
  - If you are doing the project alone, then name it with your name and last name similar to the above naming scheme.
- Submit it online on LMS (Go to the Assignments Tab) by **17:00 on October 24, 2016**.

  Late Submission Policy:

  - -10%: Submissions between 17:01 – 18:00 on the due date
  - -20%: Submissions between 18:01 – midnight (00:00) on the due date
  - -30%: Submissions which are 24 hour late.
  - -50%: Submissions which are 48 hours late.
  - Submission more than 48 hours late will not be accepted.

Grading Criteria? :

- Does it run? (Submissions that do not run will get some partial credit which will not exceed 30% of the overall project grade).
- Does it implement all the features according to the specifications and produce correct results?
- Code organization (Meaningful names, sufficient and appropriate comments, proper organization into functions and classes, clean and understandable, etc.)?
- Interview evaluation (your grade from interview will be between 0 and 1, and it will be used as a coefficient to compute your final grade. For instance, if your initial grade was 80 before the interview, and your interview grade is 0.5, then your final grade will be 80*0.5 = 40). Not showing up for the interview appointment will **result in** grade 0.

Have further questions? :

- Please contact your TAs if you have further questions. If you need help with anything, please use the office hours of your TAs and the instructor to get help.