

IS452 Midterm project information

Summary

You'll be completing a large programming problem worth your entire midterm grade. We will usually refer to this project as the Dracula project.

Due dates

1. Dracula midpoint check in:
 - a. Submit what you have done so far for the Dracula programming problem and get some feedback. There isn't a specific amount that needs to be done, but you do need to have some code for me to review. The code doesn't need to be functional! The more you have done then the more I can provide feedback on.
 - b. Also submit what you have so far (an unedited draft is fine) for your narrative. Use this document to include the questions you have for us.
 - c. This checkin step is worth 10 points of your midterm grade.
2. Final midterm due date:
 - a. All elements are due in for grading
 - b. Like your homework, each programming element will have points. You will receive points and partial credit for each. A rubric will be posted later on in the semester.

Dracula programming problem

Read these directions carefully and please ask if you have any questions about the requirements.

You should review and start pondering and working on this one throughout the beginning of the course. Parts of your regular homework assignments will be either directly applicable to this problem or strongly related. Please make use of those patterns and borrow code from your previous assignments.

You will have been exposed to everything you need to complete this task by the time of the check in due date. However, I know that you likely haven't synthesized all that knowledge together. That's why the final submission is two weeks away. Very few students will have completed the midterm at the point of checkin, and this is not the expectation.

I want you to review this project regularly and early in the semester, but remember that you will be building up your knowledge to complete the task. There will be elements you cannot solve until closer to the check in date.

A narrative with your code is also required. This is a larger problem, so this narrative should be about 400-500 words. Again, write your narrative as you work.

Use the plain text copy from Project Gutenberg already uploaded to the moodle assignment. This will ensure that there aren't any strange problems with the formatting. The original source is here: <http://www.gutenberg.org/cache/epub/345/pg345.txt>

Do not alter the original text document! We will test your code on a fresh local copy.

As usual, 4CR students will start with the 2CR tasks and continue on. Please read the entire assignment before starting to work on it.

2 credit hour students:

Your job will be to parse through this document, find the individual chapters, and write those out as separate files.




























- The file names should be “Dracula-Chapter-#” where you fill in the chapter number (where you see # there in the pattern) but as a number (e.g. 1, 2, 3, etc) and not a roman numeral.
 - For example, your file names should be Dracula-Chapter-1.txt, Dracula-Chapter-2.txt, etc.
 - No need for leading 0s.
 - **DO NOT CONVERT THE ROMAN NUMERALS.** You will not need to do anything with the roman numerals. Just use a counter starting at 1. There’s nothing meant to be tricky about this element of the task. You also won’t be able to consistently grab the chapter number from inside the chapter, so you will need to use some form of a counter.
- Each chapter file should start with “CHAPTER...” as the first line and contain exactly the text content of that chapter. Don’t worry about extra newlines at the beginning or the end, but you should not have extra newlines between the text lines (watch out for this one).
 - Example: Dracula-Chapter-1.txt should start with “CHAPTER I” and the last line of text should be “sky.” Plus maybe some newlines on either end, but no other text.
 - This means you’ll need to retain or recreate the newlines within the text file as they appear.
 - The content of your chapter files needs to look exactly as it does from the original file (but again, don’t worry about extra newlines at the beginning and the end of the file).
- You may not hard code any line numbers, position numbers, or chapter numbers into your script (as in explicitly list a line number for the text in the code. Navigating this can be tricky, which is where the check in comes in. If I see anything that looks like hard coding, I’ll tell you. I’m also happy to answer any emails to ask if the method you are using is hard coding. **If you are struggling and want to hard code these things in, you’ll only lose small points on that element.** This will let you complete the assignment so you can get points on the other elements.
- More details:
 - you can look for text that says “CHAPTER I” or “CHAPTER” but you cannot do this 27 times to find all the chapters).
 - You absolutely can use split, though, on specific text. But again, you can’t do this for each chapter.
 - If you are doing anything 27 separate times in your code, you are hard coding.
 - You can use method like .index(), .find(), etc to find the start and end positions and then slice, but you may not add these position numbers directly into your code. In other words, you can use position numbers in your code, so long as you are using a python tool to find those position numbers.

- Minimal use of .pop() or slicing is allowed. This means that you must use search strategies to detect where these things are and store the values in variables. You will need to find a way to detect where the text starts and ends (warning: there is more text after the book, which you will need to figure out how to get rid of).
- If you are looking and saying that the chapters start on line 100 and go through line 10000, you are hard coding. Use other string processing tools to break it apart

4 credit hour students:

Complete the 2 credit hour portion, but you will also need to detect the chapter titles from the beginning of the file and include them in the file names. You can find these chapter titles in the CONTENTS section of the file.

- You may not hard code these chapter names anywhere in your script. You must write something that detects them and then can match them up with the chapter numbers.
- **DO NOT CONVERT THE ROMAN NUMERALS.** Again, please just use a counter.
- You will also need to clean these titles so they can be appropriate file names.
 - Example: spaces should be replaced with _ (an underscore), and all punctuation removed.
 - So your file names should be “Dracula-Chapter-1-Jonathan_Harkers_Journal.txt, ..., Dracula-Chapter-7-Cutting_from_The_Dailygraph_8_August.txt. Later is a screen shot of a completed example.

 Dracula-Chapter-1-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-2-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-3-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-4-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-5-LettersLucy_and_Mina.txt
 Dracula-Chapter-6-Mina_Murrays_Journal.txt
 Dracula-Chapter-7-Cutting_from_The_Dailygraph_8_August.txt
 Dracula-Chapter-8-Mina_Murrays_Journal.txt
 Dracula-Chapter-9-Mina_Murrays_Journal.txt
 Dracula-Chapter-10-Mina_Murrays_Journal.txt
 Dracula-Chapter-11-Lucy_Westenras_Diary.txt
 Dracula-Chapter-12-Dr_Sewards_Diary.txt
 Dracula-Chapter-13-Dr_Sewards_Diary.txt
 Dracula-Chapter-14-Mina_Harkers_Journal.txt
 Dracula-Chapter-15-Dr_Sewards_Diary.txt
 Dracula-Chapter-16-Dr_Sewards_Diary.txt
 Dracula-Chapter-17-Dr_Sewards_Diary.txt
 Dracula-Chapter-18-Dr_Sewards_Diary.txt
 Dracula-Chapter-19-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-20-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-21-Dr_Sewards_Diary.txt
 Dracula-Chapter-22-Jonathan_Harkers_Journal.txt
 Dracula-Chapter-23-Dr_Sewards_Diary.txt
 Dracula-Chapter-24-Dr_Sewards_Phonograph_Diary_spoken_by_Van_Helsing.txt
 Dracula-Chapter-25-Dr_Sewards_Diary.txt
 Dracula-Chapter-26-Dr_Sewards_Diary.txt
 Dracula-Chapter-27-Mina_Harkers_Journal.txt

Extra credit (open to 2&4 hour students):

Write a script that calculates how many lines, words, and letters there are in each chapter, and have it write out the data as a separate data file with headers and one row of data for each chapter. You can include this code in your main py file (just make it clear in the comments) or as a separate file. Do not add the output to your chapter files. This output file should look like a CSV, with the headers at the top and each line below it lines of data. Do not add any more information or formatting to this.

Submit both your script (if separated) and the output file. Include your extra credit narrative (about 100 words) to your main narrative.

Extra credit points will be applied to your midterm score first and any remaining will be counted as weekly homework extra credit. There will be a separate “roll over” assignment for this.

Example output file(with obviously made up numbers) below. Your formatting must match this!

```
chapter, characters, words, lines
1, 100000, 10000, 1000
2, 100000, 10000, 1000
etc
```

Final submission checklist

You will need to zip these files up for submission (Moodle has a limit of 20 files).

- Narrative for the programming problem (with extra credit narrative if you’ve completed it)
- Your copy of Dracula.txt
- Your output files (the 27 chapter files)
- (and if you’ve done the extra credit...)
- Extra credit py file if you’ve made them separate
- Extra credit output file