David Hall
Project 1: Stepwise Refinement
CS333
10/26/16

Design document for Project Gutenberg Book Indexing Application

- Initialize bookObject
- Create Book
    - Create pages
        - initialize ifstream bookIn to file name given in command args
        - while not end of file
            - create empty string
            - set counter linesIn to 0
            - for linesIn < MAX_PER_PAGE AND NOT end of file, read in page
            - read each line with getline()
            - increment counter
            - Create new Page with input string
            - add to Book object
    - Find Title
        - Pull main text from first page. Find the string between the end of the substring "Title:" and the beginning of the substring "Author:".
        - Set this as the primary title
            - set the member variable of the Book object
            - loop through all pages and set as title variable of Page objects
    - Add HTML "wrapper" to page
            - Add first standard tags to one string, inserting title
            - Create the page links by using a formatted page number
                - If page number is one, only do next and index
                - If page number is the last page number only do previous and index
            - add closing tags to end of string
            - set Page html string to new built string
    - create index page
        - create two-dimensional array of index-term objects
            - 'x-axis' represents each letter,'y-axis' holds the terms in order
            - create parallel array to keep count of the number of terms per letter
                - Initialize to 0's
        - create data structure for index terms
            - Store the term, the page numbers the term occurs on at least once, and the total number of pages it occurs on
            - store a boolean flag that is true if the term is over the limit. To be used during "cleanup" phase after all terms are addedt

        - iterate through book page by page and add terms as required to array-matrix

            - create new entry if:
                - Preceded by new line OR space AND
                - Over 3 characters
            - Convert entry to lower case
            - Record page #
            - Increment a counter of how many times occured
                - increment only once per page

        - iterate through data structure and discard entries with over PAGE_THRESHOLD repeat count, using flag in data structure
- Out index page to its own HTML file
    - Create header of page with title
    - Create list of letter headers in <h2> format
    - for each letter:
        - loop through index terms and add entries for each term into <ul> list
        - add pages for each entry to <a> list of links to each page
    - add ending code
    - save file to "indexPage.html"
- Loop through book and output page HTML's

FIN

<u>Discussion on how the test spec changed:</u>

1) Create constants: This section I eliminated entirely. Because there were so many individual little

tags for each purpose and they were used in isolated locations in the code, there was no need to go to the trouble. I did do one static constant to represent the letters of the alphabet, but this was purely for readability and convenience.

2)Create Pages: Overall I was pretty faithful to this plan.

-Some things, like getting the title, proved a unique challenge because I did not account for the fact that a) reading the whole file into one string is cumbersome b)other methods involve reading in part of the file and then searching for it. Overall I found it easier to just make everything without the title and add it later.

-I used an overall Book Class to manage the higher level organization of the pages, but I still used the same order of operations on the pages.

-By the time I had done this I had already found the title. By using page numbers and processing by the page, I did not need to do much with checking for a beginning or an end

-The building of the HTML files I was pretty faithful to

-Since I was using page objects, each one contained both the "main text" which was the string of the book and the "html text" which was converted for file output

3)Create Index Page: This took me about 4 times as long to do as the first part for one ultimate file. THat being said, I followed the Design Spec almost to the T. The only significant deviation was keeping the added index terms in order and in idividual sections by letter as they were added. This did not require a significant increase in time and prevented a larger sort algorithm