Milestone #2

Team: Project_Team 26
Members: Nhi Dinh, Zhang Haozhe, Teresa Cheung, James Boultinghouse, Thien Pham, Jessie Huang, Travis Nguyen

Idea: COT3100: Discrete structures dictionary/"study guide"

- **Will be based off textbook: Rosen Discrete Mathematics and Its Applications, 7th edition and UF COT3100 lecture notes**

1) What is the system definition? Why is the system important?
    a) Our system definition will be the overall layout of our dictionary program. It is important so that we know what parameters and functions we want to write and execute with the working objects of our code.
2) What should be the inputs? What should be the outputs? What are the required flow/logic required for the proposed system?
    a) Inputs - user selections, terms, symbols (will be string values), examples
    b) Outputs - menu choices, terms and definitions (including examples), proofs
    c) Flow and Logic - linked lists, arrays, and memory handling, sorting algorithms, exception handling
3) How many modules are required?
    a) Main menu and exception handling
    b) List of terms and definitions, also include examples and sources
    c) Proofs and examples
    d) A module for all our sorting methods
    e) Memory storage (for the user to view which definitions they added) and maintenance
4) What are the classes and methods for each module/component?
    a) Main menu: options
        i) Add (to list of user definitions)
            (1) Word
            (2) Definition
            (3) Chapter to add the word
        ii) Remove
        iii) Sort
            (1) Alphabetical
            (2) By Chapter in book
                (a) By Concept
            (3) Go Back (in case user did not mean to chose sort)

iv) Previous Items (limit to 32 definitions in memory)
(1) Show dictionary at its current state
v) Find terms and definition modification
vi) Credits (sources)
vii) Quit

b) Terms and definitions: store terms
i) Most likely storing with linkedlist since our dictionary is going to be dynamic and subject to change

c) Organizational methods: sort, add, remove (linked lists)
i) Sort
(1) alphabetically
(2) chapter term was learned in
ii) Add
(1) Default: add term alphabetically
(2) While(adding: add the chapter term/concept is found in (have to prompt user for this) and use chapter as keyword for 'sort by chapter' method
iii) Remove
(1) Straightforward, just remove
(2) sort and check afterward

d) Memory storage and maintenance
i) Usage of linked lists, store current data into arrays
ii) Use loop to display data (terms and definitions)

5) What are all the shared classes/methods across all modules?
a) Dictionary module will call from organize module when sorting, adding, and/or removing terms/concepts
b) Our main module will include the dictionary and organize modules and will let the user decide what methods they want to use and what words, terms. Definitions, proofs, and examples they want to add to the dictionary/"study guide"

6) What is the best way to divide the coding task?
a) Words - The coding can be divided into the different sections of the dictionary. Depending on the number of terms and definitions, the coding task can be split up according to the letters of the alphabet or a logical set of words that are related to one another according to definition, curriculum, or difficulty.
b) By module
c) Functions - Other coding tasks (such as the options on the Main Menu), can be split up as needed.

    d) Sorting - Each member or subgroup of members can work on the organization of terms.
7) What will the makefile look like?
    a) dictionary.cpp
        i) Include user menu, main method and other methods
        ii) #dictionary.h
            (1) Stores all our terms and concepts
        iii) #organize.h
            (1) Stores all our methods for organizing our dictionary
8) Group planned deadlines to carry out implementation/testing on each module:
    a) March 25 (tentative) - a list of all of the terms and definitions we're going to use, organized layout of the modules and algorithms
    b) April 1 - have the menu created and start working on the dictionary terms
    c) April 5 - have the LinkedList stuff created
    d) April 10 - have the sorting methods created, make sure that the outputs are correct
    e) April 11 - start testing and debugging
    f) April 12 and April 19 - Go to discussion sessions to ask TAs any questions we have about debugging
    g) April 23 - project is due by 8:30am
9) Goals
    a) Create a dictionary with a friendly user interface
    b) Create an algorithm to display similar words if inputted word does not exist
10) Possible Additional Features
    a) Flashcards for studying option