Andreas Josef C. Diaz

ENGG 151.01 – A

Project 3: Linear Time-Invariant System Simulator

Documentation

- Resulting program has three source files
  - main
    - Interface of the program
    - At first, shows the program title, the names of the creators, and a hint to type "help" if the user is clueless on what to do with the program.
    - Opens the log file in append mode to write anything that would result from the user commands
    - It would go inside a loop that would take in commands from the user.
      - It would wait for the user to input commands (User-initiated)
      - The loop, and the whole program, would only stop once the user inputs the "exit" command.
    - List of commands possible follows project specifications
      - With the addition of initcond command, which inputs the initial conditions to the system.
    - main uses stringstream to parse the required inputs
    - main also has error feedback
      - If the user-input is not within the list of commands possible, the command prompt would say that it is an unknown command and would hint to use "help" command for the list of possible commands
      - If the user-input does input a possible command but has not specified a specific file (in the case of system and signal files), the program would say that there is a syntax error
      - If the files inputted are not accessible nor invalid, the program would inform the user. It would not push through with getting values with the specified file.
    - The program would not push through with certain commands if a system is not yet specified.
      - Program would inform the user that there is an error as there is no specified system
    - Informs the user if the certain commands are done successfully
      - Also tries appends the results on the log file.
    - Once the signal command is used, it will output the results of the LTI system on the command prompt only if the signal file contains 30 signal files or less.
  - system
    - getSystemParameters Function: Reads the LTI System file and gets the necessary values of M+1 and N, which are used for initializing the arrays
      - If the values are incorrect, it would return false, which would output an error message to the user

        - getSystemCoefficients Function: Reads the LTI System file and gets the LTI system coefficients.
            - If the values are incorrect, and are not the same size as the values of M and N combined, it would return false, which would output an error message to the user
        - getInitialConditions Function: Part of the initcond command, it uses the same process as getSystemCoefficients Function, but it was created for code readability.
        - computeSystem Function: Calculates the next value of the output y(n) given the past values of the input and output, and the coefficients of the LTI system. It follows the general equation for an LTI system.
    - signal
        - isValidLine Function: Checks if the line in the signal file is valid. Used in checking the first line of the signal file and the succeeding lines (which are the signal values) in the signal file.
        - getSignalValues Function: Obtains all the signal values of the signal file and places them in a vector until the end of the file or until it encounters an invalid line.
- Resulting program has 3 headers
    - Source files system and signal have their respective headers, which is used by main source file to call on functions from the two source files.
    - DEBUG header
        - Only contains #define DEBUG
        - Main purpose is to make #define DEBUG global on all source files
        - If it is not commented out, the command prompt will output certain details in the command prompt.
            - Values of all the coefficients of the current system
            - Values of all input y(n) and output x(n) once the memory is cleared or once there are initial inputs
        - It is currently commented out.
- Testing the Program
    - Please Check the files from the Test Cases folder
        - Each case, except for cases 5 and 6, will contain their own signal file, system file, spreadsheet for double checking the values, the screenshot of the program processing the inputs, and the output lti log file.
        - Cases 5 and 6 utilizes files from the first 4 cases, and so they do not need to have their own signal file, system file, and spreadsheet.
    - Case 1: When M > N, no initial conditions, and 100 input signal values
        - M+1 = 4, N = 2
        - See case1 spreadsheet.ods for the coefficients and the input signal values
        - Coefficients are generated using a random number generator
        - Input signal values generated by using sin(0.32n)
    - Case 2 – When N > M, system has initial conditions, and 100 input signal values
        - M+1 = 3, N = 4
        - See case2 spreadsheet.ods for the coefficients and the input signal values

        - Coefficients and initial conditions are generated using a random number generator
        - Input signal values generated by using cos(0.73n)
    - Case 3 – When M+1 = N, no initial conditions, and 100 small input signal values
        - M+1 = 3, N = 3
        - See case3 spreadsheet.ods for the coefficients and the input signal values
        - Coefficients are generated using a random number generator
        - Input signal values generated by using 10^{-6}sin(0.8n)
        - Checks if the program works with small values (those with E)
    - Case 4 – Large values of M+1 and N, no initial conditions, 1024 input signal values
        - Stress Test
        - M+1 = 50, N = 49
        - See case4 spreadsheet.ods for the coefficients and the input signal values
        - Coefficients are generated using a random number generator
        - Input signal values generated by using sin(0.8n)
    - Case 5 – Does case 1 first and then case 2
        - Tests consecutive cases
    - Case 6 – Does case 3 first and then case 4
        - Tests consecutive cases, especially when M and N become large.