Andreas Josef C. Diaz

ENGG 151.01 – A

**Project 2: Fourier Transform**

**Documentation**

* Resulting program has three source files
  + main
    - Interface of the program
    - Checks if the user inputs are valid
      * Checks if there are 6 or 7 arguments, else it will return a syntax error along with a guide for the correct syntax
      * If some arguments are not valid, it will inform the user
        + Signal file does not exist
        + Not a valid signal file
        + Inputted sampling frequency, start frequency, and end frequency are not positive double/s
        + nSteps is not a positive integer
      * Proceeds with Fourier transform and generating output file once inputs are valid
  + fileIO
    - isValidLine Function: First extracts each space-delimited string from the signal line then parses each argument. It also checks for extra characters
      * If signal file has a start index, it will still read it but it will not be used on other functions anymore; the start index value will be thrown out.
      * Used on readFirstLine function, readLines function.
    - generateOutputFile Function: opens the specified output file and appends the results of Fourier transform
      * Retains previous results as it does not overwrite them
      * If the user does not specify any file name to place the output results, it will default to “dftlog.txt”
  + fourierTransform
    - Global Variable pi: Used for fourierTransform function. Initialized by using the value of
    - checkArguments Function: It checks all arguments of the user inputs and informs the main source file if there is a problem with one argument. It gives a detail on what argument/s are wrong so that the main source file can output a message on command prompt.
    - readArguments Function: Turns the numerical user inputs into their respective formats (double or integer).
    - fourierTransform Function: Uses the signal values and numerical user input arguments to get the discrete Fourier transform of the signal file.
      * Calculates the Real Part, Imaginary Part, Magnitude, and Phase
      * Places each one on their respective arrays, which is used for generateOutputFile function.
* Resulting program has 3 headers
  + Source files fileIO and fourierTransform 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
    - Connected to all the source files as most functions have #ifdef DEBUG arguments
    - If it is not commented out, the command prompt will output details of the inputs in the command prompt.
      * Shows the numerical user input arguments
      * Shows all the signal values from the input signal file
      * Shows the result of the discrete Fourier transform
    - It is currently commented out so there would be no outputs on the command prompt.
* Testing the Program
  + Check the files from the Test Cases folder
    - Contains Input Signal File (.txt), Command Line Used (.png), Output Signal File (.txt), and a spreadsheet calculator (.ods)
      * The spreadsheet calculator was used to verify the program outputs for all five cases.
      * The spreadsheet was created based on the discussion last March 30, 2022
      * Some modifications on the spreadsheet:
        + Imposed a negative on the imaginary part in accordance with the definition of DFT
        + Used to get the phase
  + Case 1 – A composite signal with 14 components enabled (values come from ENGG 151.02 Lab Activity 3).
    - Fourier transform was done using 8kHz sampling rate with a frequency from 440 Hz to 480 Hz swept with 10 steps.
  + Case 2 – A normalized composite signal with 14 components enabled (values come from ENGG 151.02 Lab Activity 3).
    - Fourier transform was done using 8kHz sampling rate with a frequency from 440 Hz to 480 Hz swept with 10 steps.
  + Case 3 – A composite signal with the first 7 components enabled (values come from ENGG 151.02 Lab Activity 3)
    - Fourier transform was done using 22.1 kHz sampling rate with a frequency from 8 kHz to 10 kHz swept with 1,000 steps.
  + Case 4 – A composite signal with the last 7 components enabled (values come from ENGG 151.02 Lab Activity 3)
    - Fourier transform was done using 44.2 kHz sampling rate with a frequency from 1 Hz to 1 MHz swept with 100 steps.
  + Case 5 – Stress Test: A composite signal with 14 components enabled (values come from ENGG 151.02 Lab Activity 3). 5000 samples.
    - Fourier transform was done using 50 kHz sampling rate with a frequency from 1 Hz to 1 MHz swept with 1,000 steps.
  + Case 6 – 1 component signal enabled
    - Signal Properties
      * 0.000001 Amplitude
      * 1000 Hz frequency
      * 5000 Hz Sampling Rate
      * 100 Degree Phase
    - Generated Signal Values had E in them. This case will check if the program parses the letter E on doubles.
    - Signal file also contains starting index and a comment on the first line.
    - Fourier transform was done using 5 kHz sampling rate with a frequency from 900 Hz to 1.1 kHz swept with 10 steps.
    - Output file not specified on command line to check if it automatically generates a new file entitled “dftlog.txt”