**Documentation for PratherLogs**

**Globals**

Variables

* String fName
  + Variable that contains the first name of the student
* String lName
  + Variable that contains the last name of the student
* String[] hotkeys
  + List of Hot Key word to check inputs for
* String[] nonHotKeys
  + List of the Hot keys that won’t be logged
  + Reasons for this: If the control key is hot
* List<string> usedHotKeys
  + List of used Hot Keys
* String keyTracker
  + Makes a long list of every key pressed over a time interval
* Counters
  + Keeps track of what the log number is currently next
  + One for every log file
* Addresses
  + Variables that contain all the file paths for any file or folder that the application interacts with
  + Log Folder Addresses
    - File Paths that have to do with the student log file that would be encrypted by the application when the user is done
  + Crypt Folder Addresses
    - File Paths that have to do with the encrypted folder
* String logFile
  + Css file that will be created to assist the html files in the log folder
* String collapseFile
  + Js file that will be created to assist the html files in the log folder
* getIndexFile() returns string
  + Get function for creating the index file because it needs to be generated after student information is input
  + Returns: Index file string to be written
* getHeader(string title) returns string
  + Function that returns a html header for the log files
  + String title: Name of the title that the function will add to the header
  + Returns: Header for html file with curated title
* String htmlFoot
  + Footer for the html log files
* Bool DONE
  + Boolean to mark the project as complete
* TimeSpan timeElapsed()
  + Used to get the time elasped from the start of the project to the current time
  + Returns: TimeSpan aka time between the start of the program and the current time
* listReader(List<string> listToRead) returns List<string>
  + Method that reads a list and returns a list that contains how many times a string was repeated in a list
  + List<string> listToRead: List to be read for repeated values
  + Returns: List of repeated values and how many times they are repeated

**WriteTo**

* WriteTo.Complete()
  + Appends all the log files with a tail string to complete the html file so that it is formatted correctly
* WriteTo.writeToFile(string fileName, string write)
  + Method to write to a specific File
  + String fileName: name of the file to be written to
  + String write: string to be written to the file
* WriteTo.addDiv(string text) returns string
  + Method to replace all "\n" with "</div><div>" so that the logs are not all on one line
  + String text: string that will be modified
  + Returns: a string with </div><div> instead of \n
  + Reason for this method: When writing to the log files if the string is not modified then it will be a single line of text instead of with the intended format
* WriteTo.addPlus(string text) returns string
  + Replaces comma with a plus sign so that it can be put into a csv file
  + String text: text to be parsed for commas
  + Returns: text that had a comma now have a plus sign instead
  + Reason for this method: To put a string with a comma in a csv file without it messing up the format it we needed to replace the comma with a plus
* WriteTo.CreateFiles()
  + Method to clear the files if the exist and create the files if they don't. It will also add a head string to the log files. It also creates and adds files to the folder that are already complete files.
  + Reason for this method: We create the files we want to add to the student folder.

Write To Log Files

* WriteTo.writeToSnapshotHTML(string write)
  + Method to write to the Snapshot HTML log file
  + String write: Snapshot of the current code
* WriteTo. writeToHotKeyHTML(string write)
  + Method to write to the HotKey HTML log file
  + String write: Hot key pressed by student
  + Related: List of Hot keys in Globals
* WriteTo.writeToKeyLoggerHTML(string write)
  + Method to write to the Key Logger HTML log file
  + String write: Keys pressed by the student over an amount of time
* WriteTo.writeToClipboard(string write)
  + Method to write to the Clipboard HTML log file
  + String write: Clipboard of the student
* WriteTo.writeToOutput(string write)
  + Method to write to the Output HTML log file
  + String write: Output from when the student runs their code
* WriteTo.writeToAttention(string write)
  + Method to write to the Attention HTML log file
  + String write: Current Window that the student is looking at
* WriteTo.writeToError(string write)
  + Method to write to the Error HTML log file
  + String write: Error from the output by the student’s code

**Cryptog**

* encryptFile(string inputFile, string outputFile)
  + Encrypts a file using AES. The AES key and IV are encrypted with RSA and written to the output file for decryption later.
  + String InputFile: is the path to the file to be encrypted.
  + String OutputFile: is the path to where the encrypted file is written to.
* decryptFile(string inputFile, string outputFile)
  + Decrypts a file using AES. AES Key and IV are pulled from the begining of the file.
  + String inputFile: is the path to the file to be decrypted.
  + String outputFile: is the path to where the decrypted file is written to.
* encryptSubmit()
  + Creates a zip file from the TratherLogs folder creates encrypted file in Cryptog folder from zip file then deletes TratherLogs and TratherLogs.zip
* decrpytSubmit()
  + Select multiple files to decrypt at once and check that the decrypted version was created.
* GetFileNameAppendVariation(string fileName, string variation) returns string
  + Helper function to change name of file from file path

**Form1**

* String runProcess()
  + handles the creation, execution, and exit of command prompt process also handles redirection of user input, code output, and error messages called by runToolStripMenuItem\_Click
* appendError(string error, List<string> errorList)
  + Adds error to error list
* UpdateOutput(char line)
  + Updates the output of the console
* UpdateTime(object sender, EventArgs e)
  + Timer for log files where the functions will run every so often
* saveToolStripMenuItem\_Click(object sender, EventArgs e)
  + saves all text in the input textbox to an assignment file
* submitToolStripMenuItem\_Click(object sender, EventArgs e)
  + final save of all text in the input textbox to the assignment file then encrypts the folder containing the assigment and all logs finally it exits the program
* runToolStripMenuItem\_Click(object sender, EventArgs e)
  + asynchronous method that unlocks user input text box, calls "runProcess" sets output textbox to result returned by "runProcess" and writes necessary information to "Globals.snapshothtmlAddress", "Globals.clipboardhtmlAddress", and "Globals.outputAddress" log files
* zoomInToolStripMenuItem\_Click(object sender, EventArgs e)
  + Increase magnification of text in input text box
* zoomOutToolStripMenuItem\_Click(object sender, EventArgs e)
  + Decrease magnification of text in input text box
* zoom100ToolStripMenuItem\_Click(object sender, EventArgs e)
  + Resets magnification of text in input text box
* decryptFileToolStripMenuItem\_Click(object sender, EventArgs e)
  + Calls the decryption method from Cryptog
* Form1\_FormClosed(object sender, FormClosedEventArgs e)
  + Closes the Application
* enterInput\_Click(object sender, EventArgs e)
  + Redirects input enter by user in “userInput” to command line
* keydownrec(object sender, KeyEventArgs e)
  + Records what keys are pressed. Will record key presses to log file every update interval
* runUnitTest() returns string
  + Runs the current code and tests it against the unit test selected by the user