Pseudo-Code for Assignment 3

Joseph Davidson

\* Create a variable ‘Ans’ and set it to ‘Y’

\* Initiate a loop while Ans is equal to ‘Y’

* Create a variable ‘itworked’ and set its value to False
* Initiate a loop to run while itworked is False
  + Ask the user which file they would like to open for reading
  + Create a variable ‘NAME’ to hold the file name
  + Initiate a loop to determine if the file has the file ends with ‘.txt’
    - If it does not append .txt to the end of the file name
  + Try to open the file in read mode
    - If successful end the loop
  + If the file is not opened successfully inform the user and restart the loop
* Create a variable ‘itworked2’ and set it to False
* Initiate a loop while itworked2 is equal to False
  + Create an output file for the results named after the filename
  + Initiate a loop to ensure that the output file ends in ‘.txt’
    - If it does not end in ‘.txt’; add ‘.txt’ to the output file
  + Try to open the output file for writing
    - If successful set ‘itworked2’ end the loop
  + If the output file could not be opened
    - Create a default output file called ‘Results.txt’ and open it for writing
    - End the loop
* Print the name of the processed file and where the results will be written to on the screen
* Initiate the line counter, the counter for variables that fall within the boundaries, and the counter for variables falling outside the boundaries
* Read the first line of the input file
* Start a loop to extract data from the file and ending when an empty line is read
  + Extract the three variables from the first line (NAME, x, y)
  + Remove the quotation marks from both sides of the first variable
  + Convert the x and y variables to floats
  + Initiate a loop to analyze the first line
    - If the first variable is the lower limit
      * Create a lower limit x and lower limit y variable to hold the data
      * Create a variable xy and set it to the value of 1
      * Break the loop
    - If the first variable is the upper limit
      * Create an upper limit x and upper limit y variable to hold the data
      * create a variable xyz and set it to 1
      * Break the loop
    - if xyz and xy are greater than zero
      * print the header displaying the upper and lower limits of x and y on the screen
      * write the header displaying the upper and lower limits of x and y to the output file
      * set xyz and xy to zero
    - If the variable x is less than the lower limit of x
      * Print the x value is too low on the screen
      * Write the x value is too low to the output file
      * Add one to the line counter
      * Add one to the count of files falling outside of the limits
      * Break the loop
    - If the variable y is less than the lower limit of y
      * Print the y value is too low on the screen
      * Write the y value is too low to the output file
      * Add one to the line counter
      * Add one to the count of files falling outside of the limits
      * Break the loop
    - If the variable x is greater than the upper limit of x
      * Print the x value is too high on the screen
      * Write the x value is too high to the output file
      * Add one to the line counter
      * Add one to the count of files falling outside of the limits
      * Break the loop
    - If the variable y is greater than the upper limit of y
      * Print the y value is too high on the screen
      * Write the y value is too high to the output file
      * Add one to the line counter
      * Add one to the count of files falling outside of the limits
      * Break the loop
    - If none of the above statements are true
      * Print the values are ok
      * Write the values are ok to the output file
      * Add one to the line counter
      * Break the loop
  + Read the next line from the input file
* Close the input file
* Print the summary header
* Write the summary header to the output file
* Print how many vertices were read
* Write how many vertices were read to the output file
* Print how many points were inside the boundaries
* Write how many points were inside the boundaries to the output file
* Print how many vertices were outside the boundaries
* Write how many vertices were outside the boundaries to the output file
* Close the output file
* Ask the user if they want to process another file

\* Print “Goodbye” and close the program