Program Cover Sheet

|  |
| --- |
| Name: Nathan VanSnepson |
| Assignment: Assignment 3 |
| List any parts of the assignment that do not work/were not completed: (None) |

|  |
| --- |
| Instructor’s Comments: |
| Grade: |

Program Submission Requirements: (1) all files, zipped and uploaded to Canvas and (2) a completed cover sheet, program execution screenshots and source code printed, **stapled** and turned in during class. Failure to follow the submission requirements will result in points lost on that particular assignment.

Graphical user interface, text, application, Teams

Description automatically generated

Graphical user interface, text, application, Teams

Description automatically generated

Graphical user interface, text, application, Teams

Description automatically generated

Table

Description automatically generated

A picture containing text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Imports System.IO

Imports System.Text

Module Module1

'------------------------------------------------------------

'- File Name : Module1.vb -

'- Part of Project: Assignment3 -

'------------------------------------------------------------

' Written By: Nathan VanSnepson -

' Written On: February 2, 2022 -

'------------------------------------------------------------

'- File Purpose: -

'- -

'- This file is the main file for Project Assignment3. It -

'- contains sub Main that will be called when executed. -

'------------------------------------------------------------

'- Program Purpose: -

'- -

'- The purpose of this program is to read in a text file -

'- from the user. It then processes the words in the text -

'- file. After processing, a new file gets created with the -

'- proccessed results. The user may then choose to view the -

'- new file or quit the application. -

'------------------------------------------------------------

'- Global Variable Dictionary (alphabetically): -

'- arrInvalidChar - String array of invalid word characters -

'------------------------------------------------------------

'---------------------------------------------------------------------------------------

'--- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS ---

'--- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS ---

'--- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS --- GLOBAL CONSTANTS ---

'---------------------------------------------------------------------------------------

Const COMMAND\_WIDTH = 80

Const COMMAND\_HEIGHT = 25

Const COMMMAND\_TITLE = "Word Analysis Profiler Application"

Const WORD\_COUNT\_LENGTH = 4

Const SPACES\_PLUS\_COLON\_COUNT = 4 'Space before Colon + Colon + Space before number + Space after number

'-------------------------------------------------------------------------------------------

'--- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES ---

'--- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES ---

'--- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES --- GLOBAL STRUCTURES ---

'-------------------------------------------------------------------------------------------

'---------------------------------------------------------------------------------------

'--- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES ---

'--- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES ---

'--- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES --- GLOBAL VARIABLES ---

'---------------------------------------------------------------------------------------

Dim arrInvalidChar As String() = {".", ",", " ", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"}

'-----------------------------------------------------------------------------------

'--- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS ---

'--- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS ---

'--- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS --- SUBPROGRAMS ---

'-----------------------------------------------------------------------------------

Sub Main()

'------------------------------------------------------------

'- Subprogram Name: Main -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram is the Main sub program of the applica- -

'- tion and is called on execution. It interacts with the -

'- user and handles exiting the application. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- (none) -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'-\*intMaxWordCount - an integer to hold the count of the -

'- word with the most utilization. -

'-\*slWordList - SortedList containing a key value relationsh-

'- ip where the key is the word and the value is the number -

'- of times the word is utilizes. -

'-\*strInputFileName - String that holds the name of the inpu-

'- t file that is read in from the user. -

'-\*strOutputFileName - String that holds the name of the fil-

'- e to output to. -

'-\*strShowReportFile - String holding value from user that -

'- is used to determine if they want to view the processed -

'- file. -

'------------------------------------------------------------

Dim intMaxWordCount As Integer

Dim slWordList As New SortedList()

Dim strInputFileName As String

Dim strLongestWord As String = "" 'Default value of empty String

Dim strOutputFileName As String

Dim strShowReportFile As String

'Customizes the design of the console (e.g. color)

customizeConsole()

'Gets the file name from the user

Console.WriteLine("Please enter the path and name of the file to process:")

strInputFileName = Console.ReadLine()

Console.WriteLine()

'Attempts to process the file the user entered. If it fails then it will end the program.

If Not processFile(strInputFileName, slWordList, strLongestWord, intMaxWordCount) Then

endProgram("Failed to process file! ")

Return

End If

'Notifies user process is finished successfully and prompts for a file to output to.

Console.WriteLine("Process Completed..." & vbNewLine)

Console.WriteLine("Please enter the path and name of the report file to generate:")

strOutputFileName = Console.ReadLine()

Console.WriteLine()

'Attempts to write the processed results to a new file the user specified. If it fails it will end the program.

If Not writeFile(strOutputFileName, slWordList, strLongestWord, intMaxWordCount) Then

endProgram("Failed to create file! ")

Return

End If

'Prompts user if they want to view the new file

Console.WriteLine("Would you like to see the report file? [Y/n]")

strShowReportFile = Console.ReadLine()

Console.WriteLine()

'If user does want to see the new file write it out to user

If strShowReportFile.ToLower.Equals("y") Then

'Attempts to show report. If it fails it will end the program.

If Not showReport(strOutputFileName) Then

endProgram("Failed to show report! ")

Return

End If

'User enetered they do not want to view the new file. Ends the program

ElseIf strShowReportFile.ToLower.Equals("n") Then

endProgram("Application has completed. ")

Return

Else ' user enetered invalid input, end the program.

endProgram("Invalid input! ")

Return

End If

'The program successfully finished. Ends the program

endProgram("Application has completed. ")

End Sub

Sub endProgram(ByVal strMessage As String)

'------------------------------------------------------------

'- Subprogram Name: endProgram -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram is called when the program wants to exit -

'- and informs the user why the program is quitting. It -

'- asks user to hit any key to quit. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'-\*strMessage - String containing the quit message to output-

''- to the user. -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (none) -

'------------------------------------------------------------

Console.Write(strMessage)

Console.WriteLine("Press any key to end.")

Console.ReadKey()

End Sub

Sub customizeConsole()

'------------------------------------------------------------

'- Subprogram Name: customizeConsole -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram customizes the Conosole to a specfied -

'- design. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- (none)

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (none) -

'------------------------------------------------------------

Console.Title = COMMMAND\_TITLE

Console.BackgroundColor = ConsoleColor.White

Console.ForegroundColor = ConsoleColor.DarkBlue

Console.SetWindowSize(COMMAND\_WIDTH, COMMAND\_HEIGHT)

Console.Clear()

End Sub

Function processFile(ByVal strInputFileName As String, ByRef slWordList As SortedList, ByRef strLongestWord As String,

ByRef intMaxWordCount As Integer) As Boolean

'------------------------------------------------------------

'- Subprogram Name: processFile -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram processes the word count pf a text file -

'- and returns a boolean if processing was successful. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'-\*strInputFileName - String containing the name of the file-

'- to be read from. -

'-\*slWordList - a reference to a SortedList that contains -

'- the words and word counts from a file -

'-\*strLongestWord - a reference to a String that holds the -

'- longest word in the input file -

'-\*intMaxWordCount - a reference to an Integer that holds -

'- the maximum utilization of a word in the inputted file. -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'-\*srReadInputFile - StreamReader to allow program to read -

'- from the file. -

'-\*strInputString - String to hold the items read from the -

' StreamReader. -

'------------------------------------------------------------

Dim srReadInputFile As StreamReader

Dim strInputString As String

Try 'Try to catch any exception encountered while reading from file

If File.Exists(strInputFileName) Then

'Creates a new StreamReader to read the text file

srReadInputFile = New StreamReader(strInputFileName)

'While srReadInputFile has not read the whole file

While Not srReadInputFile.EndOfStream

'Reads tothe end of file replacing periods and commas

strInputString = srReadInputFile.ReadToEnd.Replace(".", "").Replace(",", "") 'Reads to the end of the file

'Splits strInputString where there are spaces then loops through all the words

For Each strWord As String In strInputString.Split(" ")

'Sets all words to be capitalized for comparing Strings

strWord = strWord.ToUpper()

'Removes any invalid characters from word

For Each strInvalidChar As String In arrInvalidChar

strWord = strWord.Replace(strInvalidChar, "")

Next

'If SortedList contains the word add 1 to the word count.

If slWordList.ContainsKey(strWord) Then

slWordList.Item(strWord) = slWordList.Item(strWord) + 1

'If the max utilization found is less than the current utilization, update the max utilization

If intMaxWordCount < slWordList.Item(strWord) Then

intMaxWordCount = slWordList.Item(strWord)

End If

ElseIf Not strWord.Trim.Equals("") Then 'Prevents any spaces from being added to the SortedList

'Adds a new word to the SortedList

slWordList.Add(strWord, 1)

'If the longest word founds string length is less than the current string length

'update the longest word to the new word

If strLongestWord.Length < strWord.Length Then

strLongestWord = strWord

End If

End If

Next

End While

srReadInputFile.Close() 'Closes the file when done accessing it

Return True 'Returns True reporting file read was a success

End If

Catch ex As Exception

Console.WriteLine(ex.Message) 'Reports error to console

Return False 'Returns False reporting file read was unsuccessful

End Try

Return False 'Returns False reporting file read was unsuccessful

End Function

Function writeFile(ByVal strOutputFileName As String, ByRef slWordList As SortedList, ByRef strLongestWord As String,

ByRef intMaxWordCount As Integer) As Boolean

'------------------------------------------------------------

'- Subprogram Name: writeFile -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram writes processed results to a text file. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'-\*strOutputFileName - String containing the name of the -

'- file to write to. -

'-\*slWordList - a reference to a SortedList that contains -

'- the words and word counts from a file -

'-\*strLongestWord - a reference to a String that holds the -

'- longest word in the input file -

'-\*intMaxWordCount - a reference to an Integer that holds -

'- the maximum utilization of a word in the inputted file. -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'-\*arrMaximumWord - ArrayList containing a list of words -

'- with the maximum utilization. -

'-\*arrMinimumWord - ArrayList contatining a list of words -

'- with the maximum utilization. -

'-\*bldHistogram - StringBuilder used to build a String of -

'- \*'s for the word histogram. -

'-\*bldWord - StringBuilder used to create the correct spacin-

'- g to format all the words in a neat fashion. -

'-\*bldZerps - StringBuilder user to add zereos to the front -

'- of the word count. -

'-\*dblAverageWordUtilization - Double that is the calculated-

'- average utilization of the words. -

'-\*intMaxHistogramSize - Integer that is the maximum number -

'- of \*'s to use in the histogram. -

'-\*intMinWordCount - Integer that holds the lowest count of -

'- any word. -

'-\*intWordHistogramSize - Integer containing the size of the-

'- word used to build the histogram. -

'-\*intTotalWordUsageCount - Integer containing the sum of -

'- all word utilizations. -

'-\*sqOutputFile - StreamWriter to allow program to write to -

'- a file.

'------------------------------------------------------------

Dim arrMaximumWord As New ArrayList

Dim arrMinimumWord As New ArrayList

Dim bldHistogram As StringBuilder

Dim bldWord As StringBuilder

Dim bldZeros As StringBuilder

Dim dblAverageWordUtilization As Double

Dim intMaxHistogramSize As Integer

Dim intMinWordCount As Integer = Integer.MaxValue 'Defalts to max integer value

Dim intWordHistogramSize As Integer

Dim intTotalWordUsageCount As Integer = 0 'Defailts to 0

Dim swOutputFile As StreamWriter

Try 'Try to catch any exceptions encountered while writing to file

'Checks if the file exists. If it does, delete it.

If File.Exists(strOutputFileName) Then

File.Delete(strOutputFileName)

End If

'Creates a new StreamWriter to output to a file

swOutputFile = New StreamWriter(strOutputFileName)

'Writes header of file

swOutputFile.WriteLine(vbTab & vbTab & vbTab & "Word Analysis Statistics")

swOutputFile.WriteLine()

'Writes the total unique word count to file

swOutputFile.WriteLine("There are a total of " & slWordList.Count.ToString() & " unique words encountered.")

swOutputFile.WriteLine()

'Max number of stars in histogram

intMaxHistogramSize = COMMAND\_WIDTH - strLongestWord.Length - WORD\_COUNT\_LENGTH - SPACES\_PLUS\_COLON\_COUNT

'Loops through unique words in SortedList

For Each strKey As String In slWordList.Keys

'Reinitialize the StringBuilders to erase any values already associated to them

bldZeros = New StringBuilder

bldWord = New StringBuilder

bldHistogram = New StringBuilder

'Adds the current words count to the total word usage count

intTotalWordUsageCount = intTotalWordUsageCount + slWordList.Item(strKey)

'Calculates the number of starts that needs to be written for the current word

intWordHistogramSize = (slWordList.Item(strKey) / intMaxWordCount) \* intMaxHistogramSize

'While the length of the String in StringBuilder is less than the WORD\_COUNT\_LENGTH minus the current words count length appends 0's

While bldZeros.Length < WORD\_COUNT\_LENGTH - slWordList.Item(strKey).ToString().Length

bldZeros.Append("0")

End While

'Appends words count to String builder

bldZeros.Append(slWordList.Item(strKey))

'Appends the word to StringBuilder

bldWord.Append(strKey)

'While the length of the StringBuilder is less then the length of the longest word add a space

While bldWord.ToString().Length < strLongestWord.Length

bldWord.Append(" ")

End While

'While length of StringBuilder for Histogram is less than the Histogram size for the current word append a \*

While bldHistogram.ToString().Length < intWordHistogramSize

bldHistogram.Append("\*")

End While

'Writes a new line to StreamWriter containint the word, word count, and histogram

swOutputFile.WriteLine(bldWord.ToString() & " : " & bldZeros.ToString() & " " & bldHistogram.ToString())

'If the minimum word count is greater than the current word count reinitialize the arrMinimumWord ArrayList

'to remove words with more utilization and add the new word with fewer utilization

If intMinWordCount > slWordList.Item(strKey) Then

intMinWordCount = slWordList.Item(strKey)

arrMinimumWord = New ArrayList()

arrMinimumWord.Add(strKey)

'If the current word is utilized the same amount as the minimum word add the new word to ArrayList

ElseIf intMinWordCount = slWordList.Item(strKey) Then

arrMinimumWord.Add(strKey)

End If

'If the current word is utilized the same amount as the maximum word, add the word to the ArrayList

If intMaxWordCount = slWordList.Item(strKey) Then

arrMaximumWord.Add(strKey)

End If

Next

'Finds the Average utilization of all the words

dblAverageWordUtilization = intTotalWordUsageCount / slWordList.Count

'Writes the Average Word Utilization, Highest Word Utilization, and Lowest Word Utilizaion to the StreamWriter

swOutputFile.WriteLine()

swOutputFile.WriteLine("Average Word Utilization: " & dblAverageWordUtilization)

swOutputFile.WriteLine("Highest Word Utilization: " & intMaxWordCount & " on " & arrayToCommaSeperatedString(arrMaximumWord))

swOutputFile.WriteLine("Lowest Word Utilization: " & intMinWordCount & " on " & arrayToCommaSeperatedString(arrMinimumWord))

swOutputFile.WriteLine()

'Writes the StreamReader to the file

swOutputFile.Flush()

'Closes the file

swOutputFile.Close()

'Successful write to file

Return True

Catch ex As Exception

Console.WriteLine(ex.Message) 'Reports error to console

Return False 'Returns False reporting file write was unsuccessful

End Try

Return False 'Returns False reporting file write was unsuccessful

End Function

Function showReport(ByVal strFileName As String) As Boolean

'------------------------------------------------------------

'- Subprogram Name: showReport -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- This subprogram reads from a file and outputs the file to-

'- the console. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'-\*strFileName - String that has the file name to read from -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'-\*srReadInputFile - StreamReader to allow program to read -

'- from the file. -

'------------------------------------------------------------

Try 'Try to catch any exceptions encountered while reating from file

Dim srReadInputFile As New StreamReader(strFileName)

'Writes the file to the Console

Console.WriteLine(srReadInputFile.ReadToEnd())

'Closesthe file

srReadInputFile.Close()

'Successfully reported file to user

Return True

Catch ex As Exception

Return False 'Failed to read from file

End Try

Return False 'Failed to read from file

End Function

Function arrayToCommaSeperatedString(ByVal arrStringArray As ArrayList) As String

'------------------------------------------------------------

'- Subprogram Name: arrayToCommaSeperatedString -

'------------------------------------------------------------

'- Written By: Nathan VanSnepson -

'- Written On: February 2, 2022 -

'------------------------------------------------------------

'- Function Purpose: -

'- This Fucntion turns an ArrayList of Strings to a comma -

'- separated list and returns it as a String. -

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'-\*arrStringArray - ArrayList of Strings -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'-\*bldCommaList - StringBuilder to create a comma separated -

'- list of words. -

'-\* chTrimList - Char array that contains characters to remo-

'- ve from the end of the comma separated list. -

'------------------------------------------------------------

Dim bldCommaList As New StringBuilder

Dim chTrimList As Char() = {" "c, ","c}

'Loops through each string in ArrayList

For Each strWord As String In arrStringArray

bldCommaList.Append(strWord & ", ")

Next

'removes specified end characters and Returns the comma separated string

Return bldCommaList.ToString().TrimEnd(chTrimList)

End Function

End Module