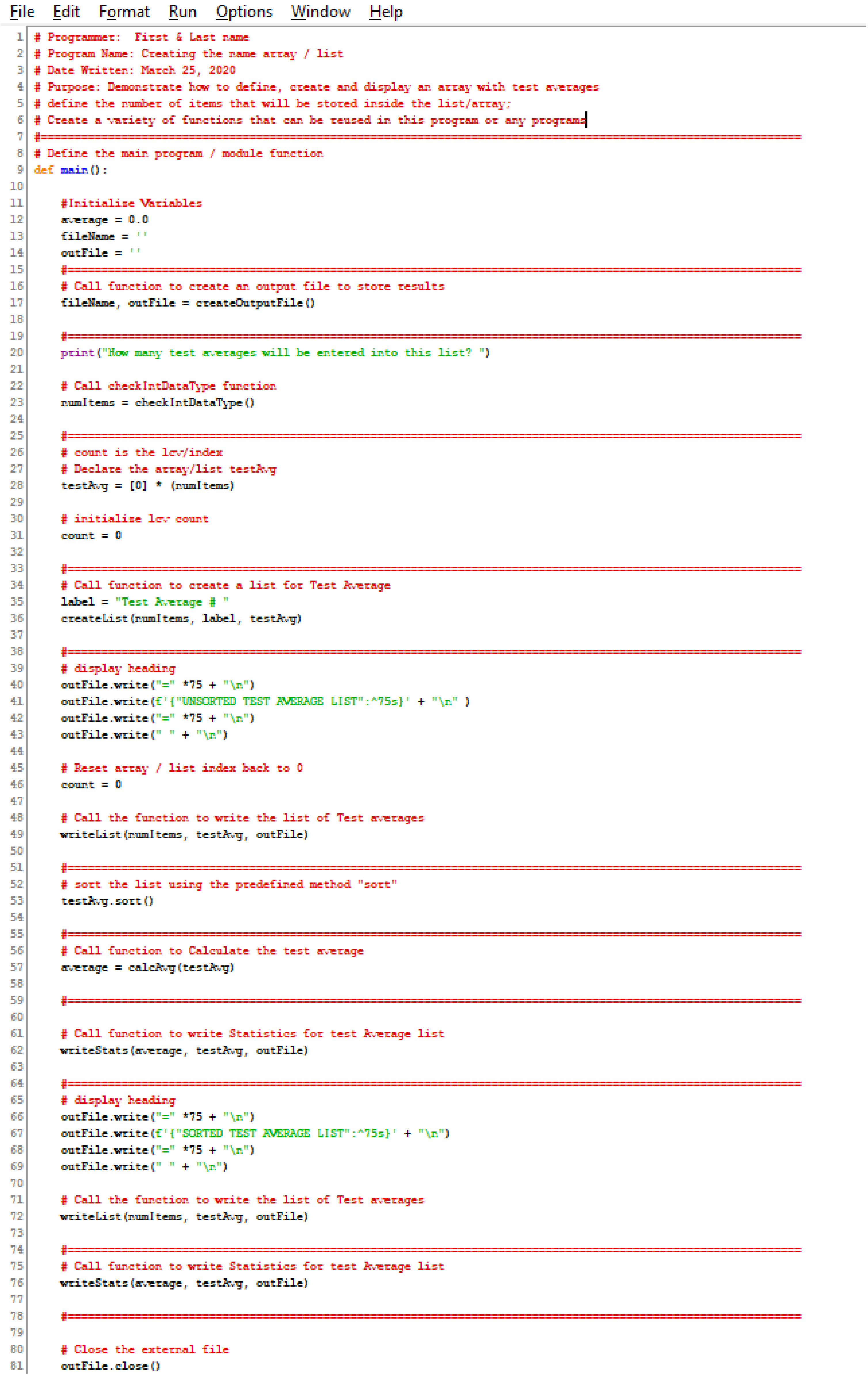
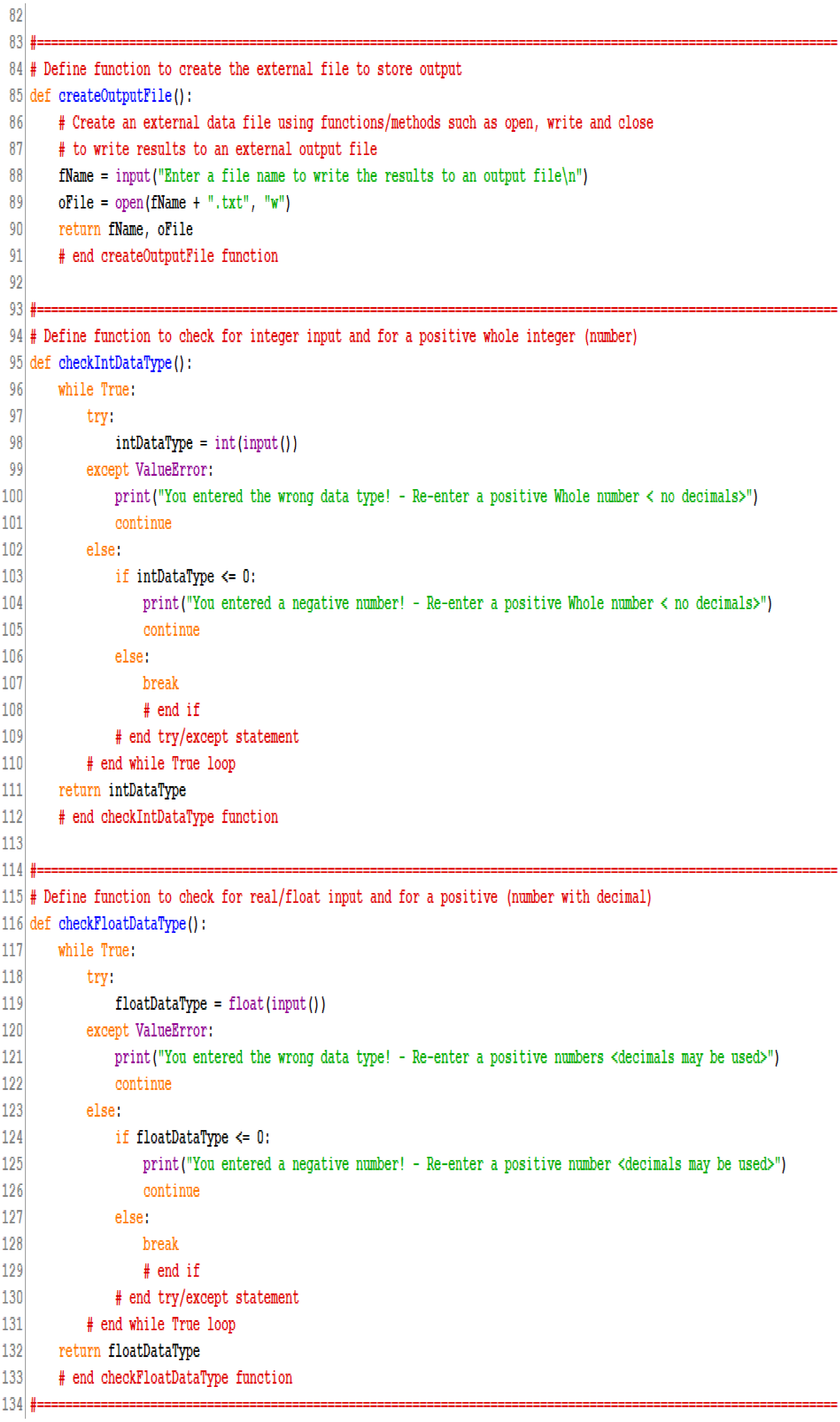
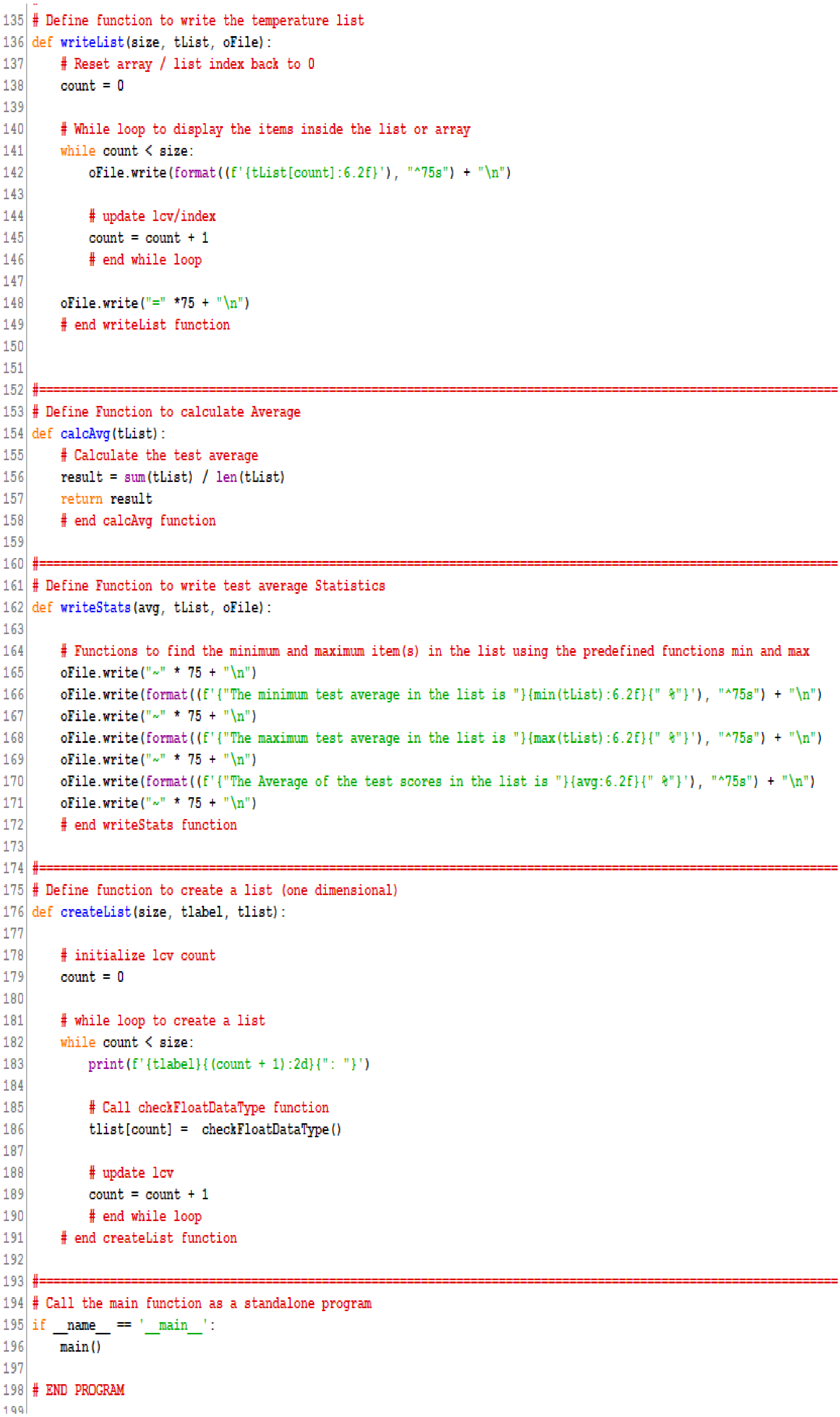
SAMPLE TEST AVG PROGRAM ORGANIZED AS FUNCTIONS:



```python
# Programmer:  First & Last name
# Program Name: Creating the name array / list
# Date Written: March 25, 2020
# Purpose: Demonstrate how to define, create and display an array with test averages
# define the number of items that will be stored inside the list/array;
# Create a variety of functions that can be reused in this program or any programs
#=================================================================================================================
# Define the main program / module function
def main():

    #Initialize Variables
    average = 0.0
    fileName = ''
    outFile = ''
    #=============================================================================================================
    # Call function to create an output file to store results
    fileName, outFile = createOutputFile()

    #=============================================================================================================
    print("How many test averages will be entered into this list? ")

    # Call checkIntDataType function
    numItems = checkIntDataType()

    #=============================================================================================================
    # count is the lcv/index
    # Declare the array/list testAvg
    testAvg = [0] * (numItems)

    # initialize lcv count
    count = 0

    #=============================================================================================================
    # Call function to create a list for Test Average
    label = "Test Average # "
    createList(numItems, label, testAvg)

    #=============================================================================================================
    # display heading
    outFile.write("=" *75 + "\n")
    outFile.write(f'{"UNSORTED TEST AVERAGE LIST":^75s}' + "\n" )
    outFile.write("=" *75 + "\n")
    outFile.write(" " + "\n")

    # Reset array / list index back to 0
    count = 0

    # Call the function to write the list of Test averages
    writeList(numItems, testAvg, outFile)

    #=============================================================================================================
    # sort the list using the predefined method "sort"
    testAvg.sort()

    #=============================================================================================================
    # Call function to Calculate the test average
    average = calcAvg(testAvg)

    #=============================================================================================================

    # Call function to write Statistics for test Average list
    writeStats(average, testAvg, outFile)

    #=============================================================================================================
    # display heading
    outFile.write("=" *75 + "\n")
    outFile.write(f'{"SORTED TEST AVERAGE LIST":^75s}' + "\n")
    outFile.write("=" *75 + "\n")
    outFile.write(" " + "\n")

    # Call the function to write the list of Test averages
    writeList(numItems, testAvg, outFile)

    #=============================================================================================================
    # Call function to write Statistics for test Average list
    writeStats(average, testAvg, outFile)

    #=============================================================================================================

    # Close the external file
    outFile.close()
```



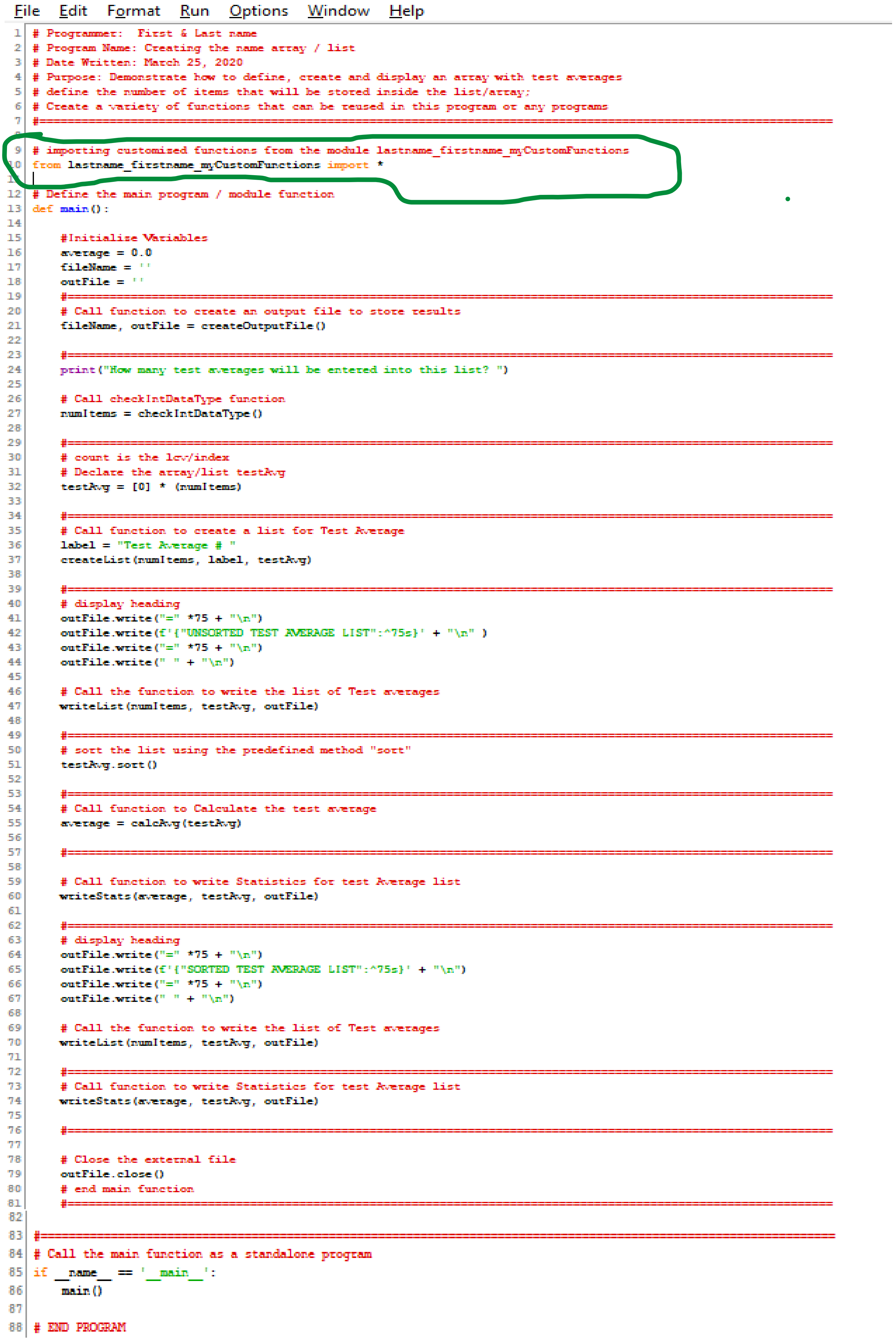
```python

#===================================================================================================================
# Define function to create the external file to store output
def createOutputFile():
    # Create an external data file using functions/methods such as open, write and close
    # to write results to an external output file
    fName = input("Enter a file name to write the results to an output file\n")
    oFile = open(fName + ".txt", "w")
    return fName, oFile
    # end createOutputFile function

#===================================================================================================================
# Define function to check for integer input and for a positive whole integer (number)
def checkIntDataType():
    while True:
        try:
            intDataType = int(input())
        except ValueError:
            print("You entered the wrong data type! - Re-enter a positive Whole number < no decimals>")
            continue
        else:
            if intDataType <= 0:
                print("You entered a negative number! - Re-enter a positive Whole number < no decimals>")
                continue
            else:
                break
                # end if
            # end try/except statement
        # end while True loop
    return intDataType
    # end checkIntDataType function

#===================================================================================================================
# Define function to check for real/float input and for a positive (number with decimal)
def checkFloatDataType():
    while True:
        try:
            floatDataType = float(input())
        except ValueError:
            print("You entered the wrong data type! - Re-enter a positive numbers <decimals may be used>")
            continue
        else:
            if floatDataType <= 0:
                print("You entered a negative number! - Re-enter a positive number <decimals may be used>")
                continue
            else:
                break
                # end if
            # end try/except statement
        # end while True loop
    return floatDataType
    # end checkFloatDataType function
#===================================================================================================================
```



```python
# Define function to write the temperature list
def writeList(size, tList, oFile):
    # Reset array / list index back to 0
    count = 0

    # While loop to display the items inside the list or array
    while count < size:
        oFile.write(format((f'{tList[count]:6.2f}'), "^75s") + "\n")

        # update lcv/index
        count = count + 1
        # end while loop

    oFile.write("=" *75 + "\n")
    # end writeList function


#=================================================================================================================
# Define Function to calculate Average
def calcAvg(tList):
    # Calculate the test average
    result = sum(tList) / len(tList)
    return result
    # end calcAvg function

#=================================================================================================================
# Define Function to write test average Statistics
def writeStats(avg, tList, oFile):

    # Functions to find the minimum and maximum item(s) in the list using the predefined functions min and max
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The minimum test average in the list is "}{min(tList):6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The maximum test average in the list is "}{max(tList):6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The Average of the test scores in the list is "}{avg:6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    # end writeStats function

#=================================================================================================================
# Define function to create a list (one dimensional)
def createList(size, tlabel, tlist):

    # initialize lcv count
    count = 0

    # while loop to create a list
    while count < size:
        print(f'{tlabel}{(count + 1):2d}{": "}')

        # Call checkFloatDataType function
        tlist[count] =  checkFloatDataType()

        # update lcv
        count = count + 1
        # end while loop
    # end createList function

#=================================================================================================================
# Call the main function as a standalone program
if __name__ == '__main__':
    main()

# END PROGRAM
```

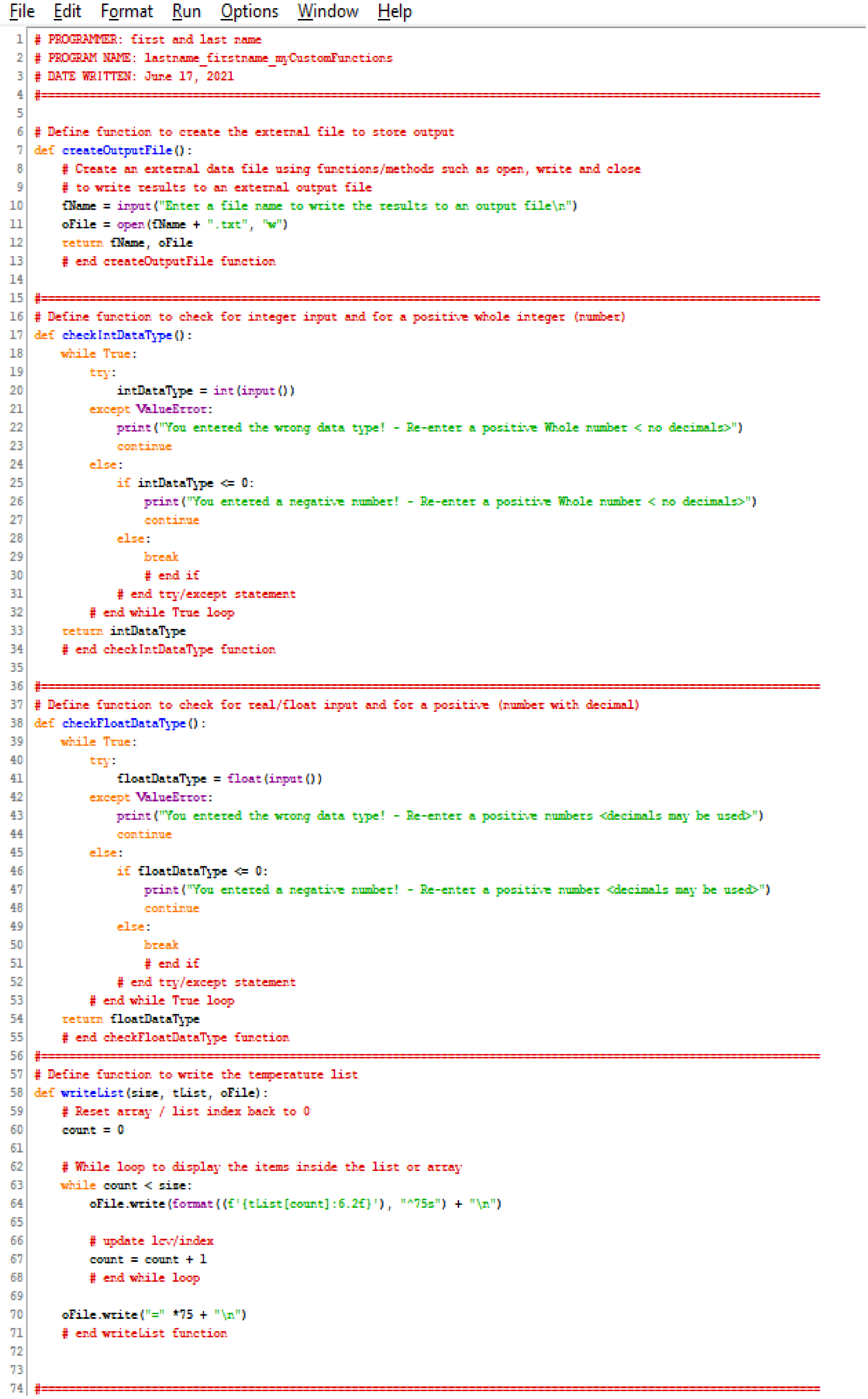
SAMPLE TESTAVG PROGRAM SEPARATED AS THE SOURCE CODE PROGRAM AND A CUSTOMIZED PROGRAM WITH FUNCTIONS:

SOURCE CODE PROGRAM:

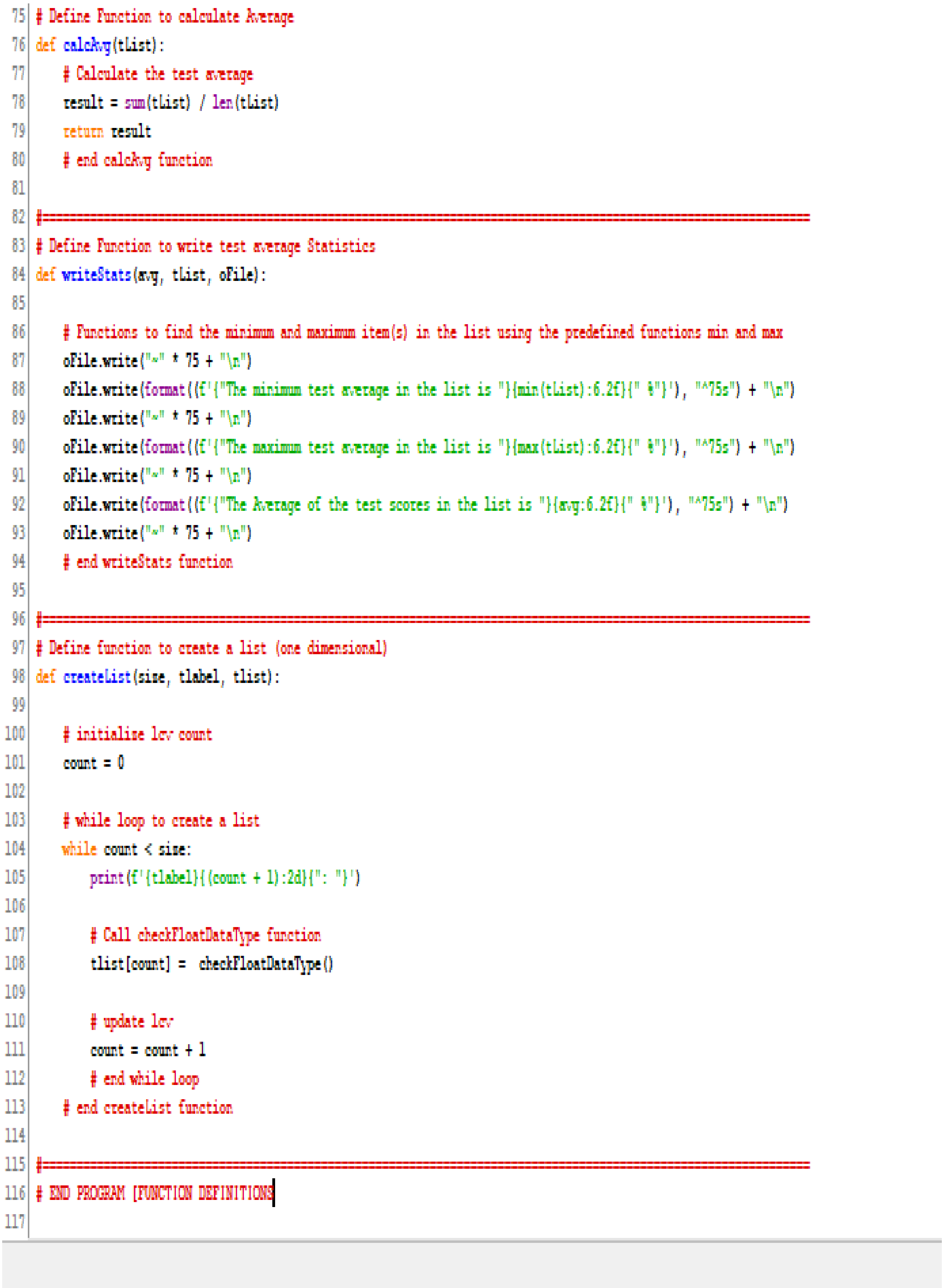


```python
# Programmer:  First & Last name
# Program Name: Creating the name array / list
# Date Written: March 25, 2020
# Purpose: Demonstrate how to define, create and display an array with test averages
# define the number of items that will be stored inside the list/array;
# Create a variety of functions that can be reused in this program or any programs
#==============================================================================================================

# importing customized functions from the module lastname_firstname_myCustomFunctions
from lastname_firstname_myCustomFunctions import *

# Define the main program / module function
def main():

    #Initialize Variables
    average = 0.0
    fileName = ''
    outFile = ''
    #==========================================================================================================
    # Call function to create an output file to store results
    fileName, outFile = createOutputFile()

    #==========================================================================================================
    print("How many test averages will be entered into this list? ")

    # Call checkIntDataType function
    numItems = checkIntDataType()

    #==========================================================================================================
    # count is the lcv/index
    # Declare the array/list testAvg
    testAvg = [0] * (numItems)

    #==========================================================================================================
    # Call function to create a list for Test Average
    label = "Test Average # "
    createList(numItems, label, testAvg)

    #==========================================================================================================
    # display heading
    outFile.write("=" *75 + "\n")
    outFile.write(f'{"UNSORTED TEST AVERAGE LIST":^75s}' + "\n" )
    outFile.write("=" *75 + "\n")
    outFile.write(" " + "\n")

    # Call the function to write the list of Test averages
    writeList(numItems, testAvg, outFile)

    #==========================================================================================================
    # sort the list using the predefined method "sort"
    testAvg.sort()

    #==========================================================================================================
    # Call function to Calculate the test average
    average = calcAvg(testAvg)

    #==========================================================================================================

    # Call function to write Statistics for test Average list
    writeStats(average, testAvg, outFile)

    #==========================================================================================================
    # display heading
    outFile.write("=" *75 + "\n")
    outFile.write(f'{"SORTED TEST AVERAGE LIST":^75s}' + "\n")
    outFile.write("=" *75 + "\n")
    outFile.write(" " + "\n")

    # Call the function to write the list of Test averages
    writeList(numItems, testAvg, outFile)

    #==========================================================================================================
    # Call function to write Statistics for test Average list
    writeStats(average, testAvg, outFile)

    #==========================================================================================================

    # Close the external file
    outFile.close()
    # end main function
    #==========================================================================================================

#==============================================================================================================
# Call the main function as a standalone program
if __name__ == '__main__':
    main()

# END PROGRAM
```

lastname_firstname_MyCustom_Functions program containing the functions.



```python
# PROGRAMMER: first and last name
# PROGRAM NAME: lastname_firstname_myCustomFunctions
# DATE WRITTEN: June 17, 2021
#==========================================================================================

# Define function to create the external file to store output
def createOutputFile():
    # Create an external data file using functions/methods such as open, write and close
    # to write results to an external output file
    fName = input("Enter a file name to write the results to an output file\n")
    oFile = open(fName + ".txt", "w")
    return fName, oFile
    # end createOutputFile function

#==========================================================================================
# Define function to check for integer input and for a positive whole integer (number)
def checkIntDataType():
    while True:
        try:
            intDataType = int(input())
        except ValueError:
            print("You entered the wrong data type! - Re-enter a positive Whole number < no decimals>")
            continue
        else:
            if intDataType <= 0:
                print("You entered a negative number! - Re-enter a positive Whole number < no decimals>")
                continue
            else:
                break
                # end if
            # end try/except statement
        # end while True loop
    return intDataType
    # end checkIntDataType function

#==========================================================================================
# Define function to check for real/float input and for a positive (number with decimal)
def checkFloatDataType():
    while True:
        try:
            floatDataType = float(input())
        except ValueError:
            print("You entered the wrong data type! - Re-enter a positive numbers <decimals may be used>")
            continue
        else:
            if floatDataType <= 0:
                print("You entered a negative number! - Re-enter a positive number <decimals may be used>")
                continue
            else:
                break
                # end if
            # end try/except statement
        # end while True loop
    return floatDataType
    # end checkFloatDataType function
#==========================================================================================
# Define function to write the temperature list
def writeList(size, tList, oFile):
    # Reset array / list index back to 0
    count = 0

    # While loop to display the items inside the list or array
    while count < size:
        oFile.write(format((f'{tList[count]:6.2f}'), "^75s") + "\n")

        # update lcv/index
        count = count + 1
        # end while loop

    oFile.write("=" *75 + "\n")
    # end writeList function


#==========================================================================================
```



```
# Define Function to calculate Average
def calcAvg(tList):
    # Calculate the test average
    result = sum(tList) / len(tList)
    return result
    # end calcAvg function

#==========================================================================================================================
# Define Function to write test average Statistics
def writeStats(avg, tList, oFile):

    # Functions to find the minimum and maximum item(s) in the list using the predefined functions min and max
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The minimum test average in the list is "}{min(tList):6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The maximum test average in the list is "}{max(tList):6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    oFile.write(format((f'{"The Average of the test scores in the list is "}{avg:6.2f}{" %"}'), "^75s") + "\n")
    oFile.write("~" * 75 + "\n")
    # end writeStats function

#==========================================================================================================================
# Define function to create a list (one dimensional)
def createList(size, tlabel, tlist):

    # initialize lcv count
    count = 0

    # while loop to create a list
    while count < size:
        print(f'{tlabel}{(count + 1):2d}{": "}')

        # Call checkFloatDataType function
        tlist[count] =  checkFloatDataType()

        # update lcv
        count = count + 1
        # end while loop
    # end createList function

#==========================================================================================================================
# END PROGRAM [FUNCTION DEFINITIONS

```

MAKE SURE THAT THE SOURCE CODE PROGRAM, TESTAVG is stored in the same folder were the lastname_firstname_myCustomFunctions.py are stored. BOTH MUST BE INSIDE THE SAME FOLDER.