Table of Contents

[Project Overview 2](#__RefHeading___Toc4258_1370385848)

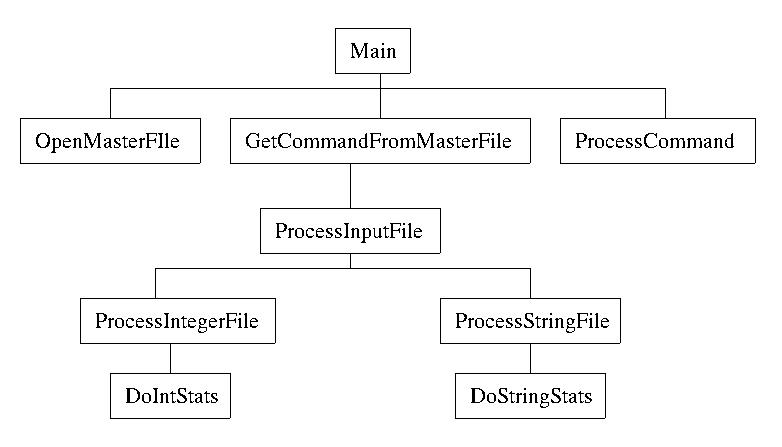
[Top Level Functions 3](#__RefHeading___Toc4260_1370385848)

# Project Overview

The purpose of this project is to write a program which can compute simple statistics about several files. The project will read commands from a user supplied master file, perform the computation and print the output to the screen.

There are multiple error conditions to consider.

**Structure Chart**



# Top Level Functions

The main function needs to control the master file. Opening, retrieving information and closing. Responsibilities for processing the sub files will be deferred to another set of functions.

**Main**

Narrative: The routine should open the master file, read the commands and pass these to the appropriate routine.

*Open the master file.*

*Get a command from the master file*

*while not at the end of the commands*

*Process the command*

*Get a command from the master file*

*Close the master file.*

**Function:** OpenMasterFile

Narrative: This function will request the file name from the user and open the master file. If there is an error , this will be detected.

Input: none

Output: A input file stream, either open or in error

*Prompt the user for the master file name*

*Read in the name*

*Attempt to open the file*

*if not successful*

*Print error message*

**Function:** GetCommandFromMasterFile

Narrative: This function will read a single line from the master file.

Input: The open master file stream

Output: The file to process, a string,

The type of the file, a string

The type of operation, a character

*Read the file name from the master file.*

*Read the type from the master file*

*Read the operation from the master file*

**Function:** ProcessCommand

Narrative: This is the high level routine to process a command. It will attempt to open the data file. If successful, it will call the routine to process the file, otherwise it will print an error message.

Input: The file name, a string

The file type, a string

The operation, a character.

Output: None

*Open the input file*

*If success*

*Process the input file*

*else*

*print an error.*

**Function:** ProcessInputFile

Narrative: This will decide the intermediate level function to call

Input: An open input file, a file stream

The file type, a string

The operation, a char

Output: None

*If type is Integer*

*ProcessIntegerFile*

*else if type is String*

*ProcessStringFile*

*else*

*report an error*

**Function:** ProcessIntegerFile

Narrative: This function will decide what low level integer function to call.

Input: An open file stream

A character representing the operation to perform

Output: none

DoIntStats

If the size is 0, print error

else

*If the operation is count*

*Print count*

*if the operation is max*

*Print max*

*if the operation is min*

*PrintMin*

*if the operation is average*

*Print sum/count*

*if the operation is sum*

*Print sum*

*else*

*print error function unknown.*

**Function:** DoIntStats

Narrative: This function computes all of the stats on an integer file

Input: An open file stream

Output: The count of the data in the file, an integer

The sum of the data in the file, an integer

The maximum value in the file, and integer

The minimum value in the file, an integer

*read in the first value*

*max = first value*

*min= first value*

*sum = first value*

*count = 0*

*while the read was successful*

*increment count*

*max = maximum(max, value)*

*min = minimum (min, value)*

*sum += value*

*read in the next value*

**Function:** DoStringStats

Narrative: Same as DoIntStats, but will not compute sum. It will compute longest and shortest.

Input:

Output:

*See above*

**Function:** ProcessStringFile

Narrative: Same as ProcessIntFile but will use strings.

Input:

Output:

*SeeAbove*