**ECE 103 Engineering Programming**

**HW-X Program Design Report**

**July 25, 2021**

**Jane Doe**

[**janedoe@pdx.edu**](mailto:janedoe@pdx.edu)

**Problem Description:**

A quality control inspection will often take a random sample of products and measure various

specifications. They use that data to make generalizations about whether the manufacturing

process is successfully meeting specifications.

This program will analyze data from a quality control inspection. The

quality control employees have taken a random sample and written down measurements, which

they saved in a text file with a specific format.

Program will read data from the text file and analyze it. The program will display a

report on screen and also write the report to a text file.

**Design Approach**

*Inputs & Outputs*

· There is only one Input (Name of text file).

· The Nine outputs are (Title with current Date, Number of sprockets, Batch is gone or not go, percentage of sprockets go, percentage of sprockets not go, Mean Hole diameter, Range of outer diameter, Maximum outer diameter, Minimum outer diameter.

*Assumptions*

· Error checking is needed for the entered filename. If file not exist then terminate.

*Method*

· Take inputs from user in the string the filename.

· Calculate the size if batch by using getCount(), it will return size of array of Measure.

· Read file line by line, parse each line of file and then store comma separated values in the array of struct.

· Get current date of system, and write DECLARE file and PRINT on the console.

· Determine if the entire batch is a Go or a No Go. Call the function isGO (), if return true then PRINT Go and if return false then not GO.

· Calculate hole diameter by calling function getMeanHoleDia(), and pass array of struct and size, it will return hole mean diameter.

· Calculate Max outer diameter by calling function getMaxOuterDia(), and pass array of struct and size, it will return Maximum of outer diameter.

· Calculate Min outer diameter by calling function getMinOuterDia(), and pass array of struct and size, it will return Minimum of outer diameter.

· Calculate the range of outer diameter by subtracting min from max outer diameter.

· Write all the data to report and PRINT on the console too.

**Logic Flow (pseudocode)**

**PROGRAM**

**MEASURE**

**Declare BOOLEAN go**

**Declare DECLARE  outerDiameter**

**Declare DECLARE holeDiameter**

**Declare DECLARE thinkness  
 END MEASURE**

**FUNCTION getCounts (fileName)**

**SET Count = 0**

**OPEN(FILENAME)**

**Line = “ “**

**WHILE (READ(Line))**

**Count++**

**ENDWHILE**

**CLOSE(FILENAME)**

**RETURN count – 2**

**END FUNCTION**

**FUNCTION getMonName(mon)**

**IF mon == 1:**

**return "January";**

**ENDIF**

**IF mon == 2:**

**return "February";**

**ENDIF**

**IF mon == 3:**

**return "March";**

**ENDIF**

**IF mon == 4:**

**return "April";**

**ENDIF**

**IF mon == 5:**

**return "May";**

**ENDIF**

**IF mon == 6:**

**return "June";**

**ENDIF**

**IF mon == 7:**

**return "July";**

**ENDIF**

**IF mon == 8:**

**return "August";**

**ENDIF**

**IF mon == 9:**

**return "September";**

**ENDIF**

**IF mon == 10:**

**return "October";**

**ENDIF**

**IF mon == 11:**

**return "November";**

**ENDIF**

**IF mon == 12:**

**return "December";**

**ENDIF**

**ENDFUNCTION**

**FUNCTION  getMeanWholeDia()**

**DECLARE sum = 0;**

**for (i=0 : size)**

**sum += measure[i].holeDiameter;**

**ENDFOR**

**return sum / size;**

**END FUNCTION**

**FUNCTION getGo()**

**Declare  go = 0**

**for (i=0: size)**

**IF measure[i].go == true)**

**go++**

**ENDIF**

**ENDFOR**

**return (go \* 100) / size**

**ENDFUNCTION**

**FUNCTION getMaxOuterDia( )**

**Declare max = -2333.0**

**for (i = 0: size)**

**IF (measure[i].outerDiameter > max)**

**SET max = measure[i].outerDiameter;**

**ENDIF**

**ENDFOR**

**return max**

**ENDFUNCTION**

**FUNCTION getMinOuterDia( )**

**DECLARE min = 233333;**

**for (i = 0 : size)**

**IF (measure[i].outerDiameter < min)**

**SET  min = measure[i].outerDiameter;**

**ENDIF**

**ENDFOR**

**return min;**

**ENDFUNCTION**

**FUNCTION  isGO( )**

**for (i = 0 : size)**

**IF (measure[i]. go == false)**

**return false;**

**ENDIF**

**ENDFOR**

**return true;**

**ENDFUNCTION**

**START main ()**

**{**

**PRINT("Welcome to the Sprocket Quality Control Analysis Tool")**

**PRINT ("Please enter the name of the file which contains the data: ")**

**DECLARE fileName = " ";**

**INPUT (fileName);**

**DECLARE size = getCounts(fileName)**

**struct Measure \*measure[size]**

**FILE file = OPENFILE (fileName, "r")**

**IF (! file NOT FOUND)**

**PRINTf (" Unable to open  fileName)**

**return -1**

**ENDIF**

**DECLARE name**

**DECLARE date**

**GETFROMFILE(name)**

**GETFROMFILE (date)**

**DECLARE sub1, sub2, sub3**

**DECLARE mon**

**DECLARE days**

**DECLARE years**

**DECLARE line**

**DECLARE val1**

**DECLARE val2**

**DECLARE val3**

**DECLARE val4**

**DECLARE a = 0**

**START WHILE (GETFROMFILE(Line))**

**{**

**READ (line, &val1, &val2, &val3, &val4)**

**measure[a].holeDiameter = val2**

**measure[a]. outerDiameter = val3;**

**measure[a].thinkness = val4;**

**IF (val1 == 1)**

**measure[a]. go = true**

**else**

**measure[a]. go = false**

**ENDIFELSE**

**a++;**

**ENDWHILE**

**DECLARE perGo = getGo (measure, size)**

**DECLARE NoGo = 100 - perGo**

**CLOSEFILE (file);**

**DECLARE day = CURRENTDATE.day**

**DECLARE month = CURRENTDATE.month**

**DECLARE year = CURRENTDATE.year + 1900**

**DECLARE report = "report\_0";**

**report = report + mon + day + year + ".txt"**

**FILE = OPENFILE (report)**

**DECLARE firstLine[200] = "Analysis of "**

**Firstline = firstLine + name +  " measured on " + getMonName(month)) + rday + ryear**

**PRINT (firstLine)**

**DECLARE second**

**Second = “ Sprockets are analysed"**

**WRITETOFILE (firstLine)**

**WRITETOFILE (second)**

**PRINT( second)**

**IF (isGO (measure, size))**

**WRITETOFILE ("The Batch is a Go")**

**PRINT ("The Batch is a Go")**

**else**

**WRITETOFILE ("The Batch is a No Go")**

**PRINT ("The Batch is a No Go")**

**ENDIFELSE**

**DECLARE ygo =  perG +  "% of sprockets are Go")**

**WRITETOFILE (ygo);**

**PRINT(ygo);**

**DECLARE ngo = NoGo  + "% of sprockets are No Go"**

**WRITETOFILE( ngo);**

**PRINTf(ngo);**

**DECLARE meanstr= "Mean Hole Diameter: "**

**DECLARE w = getMeanWholeDia (measure, size))**

**Meanstr = meanstr +  w + meanstr, " inches."**

**WRITETOFILE(meanstr)**

**PRINT( meanstr)**

**DECLARE ranstr = "Range of Outer Diameters: "**

**DECLARE z = getMaxOuterDia (measure, size) - getMinOuterDia (measure, size))**

**Ranstr = ranstr + z +  " inches."**

**WRITETOFILE( ranstr);**

**PRINT( ranstr)**

**DECLARE maxstr[100] = "\tMaximum Outer Diameter: ";**

**DECLARE x = getMaxOuterDia (measure, size)**

**maxstr = maxstr +  x +  " inches."**

**WRITETOFILE (maxstr);**

**PRINT( maxstr);**

**DECLARE minstr =  "Minimum Outer Diameter: "**

**minstr = minstr +  y + minstr + " inches."**

**PRINT( minstr);**

**CLOSEFILE(fp)**

**ENDPROGRAM**