Project 6 REPORT

**Definition:**

The purpose of the program is to read user data and use it to calculate projected salary raises and final evaluations to finally write all of the information in an understandable organized fashion. The input is the user’s input from an input file. The program will compute final weighted evaluation for each year; Total Final Weighted Evaluation; Average Final Weighted Evaluation; Salary raise %; Salary raise in $; and Salary with raise in $. All data gathered from input file will be printed to the output file with labels of what the information resembles. All data calculated (final weighted evaluation for each year; Total Final Weighted Evaluation; Average Final Weighted Evaluation; Salary raise %; Salary raise in $; and Salary with raise in $) will be printed to the output file with labels to denotate the meaning of the information. Warning messages and congrats messages will be printed depending on the user’s Average Final Weighted Evaluation score.

**Algorithm:**

Declare global variables (num of employees, num of semesters, num of years)

Declare struct address (containing string variables street, city, state, zip)

Declare struct semeval (containing double array eval[9])

Declare struct comp\_eval (containing double array compeval[3]; double variables total and average)

Declare struct Employee (containing string array header; string variables e\_name, s\_name, e\_id, e\_tele; address variable address; semeval variable e\_evals; comp\_eval variable comp\_e\_evals; double variables salary, salraise, salraiseindoll, newsal; string variable arrays letter[2], note[2])

Prototype functions (inputData; validateData; FEW; TAFWE; SRinPercent; Salary; letter; report)

Begin function main

Declare input file variable (fin)

Declare output file variable (fout)

Open input file

Open output file

Declare Employee datatype array (emp[num of employees])

Declare int variable validation (used to break/continue when error is detected)

Begin employee number loop

Call inputData function by setting variable validation equal to inputData/input function (if inputData returns a value, program will continue/break employee number loop)

Call Final weighted evaluation function (FWE)

Call Total and Average Final Weighted Evaluation function (TAFWE)

Call Salary Raise in Percent function (SRinPercent)

Call Salary function (Salary)

Call Report/Output function (Report)

End of employee number loop

Close input file and output files

Define inputData function

Declare int variable validate (used to detect errors in input)

Read Warning/Congrats message from input file (ONLY READS ONCE by using if statements and store in emp[num of emp].letter[])

Read Employee information from input file using getline (storing information in Employee datatype array emp[num of emp])

Use for loop to efficiently read employee evaluations from input file (storing information in Employee datatype array emp[num of emp].e\_evals\_eval[])

Read employee salary from input file (storing information in Employee datatype array emp[num of emp].salary)

Read employee discrepancy note from input file using getline (storing information in Employee datatype array emp[num of emp].note[])

Call validateData function by setting variable validate equal to validateData function (if validateData returns a number an error will be detected)

End of inputData definition

Define validateData function

Test length of strings stored in Employee datatype array emp[num of emp] (using if statements; return a number if error = true)

Test value of numerical data stored in Employee datatype array emp[num of emp] (using if statements; return a number if error = true)

End of validateData definition

Define FWE function

Declare weight constants for each semester

Calculate Final Weighted Evaluation for each year (by multiplying info from Employee datatype array emp[num of emp].e\_evals.eval[semester num] by semester weights and adding each semester for each year)

End of FWE definition

Define TAFWE function­­­­­­­­­­­

Calculate Total Final Weighted Evaluation (adding every year’s Final Weighted Evaluation; store it in Employee datatype array emp[num of emp].comp\_e\_evals.total)

Calculate Average Final Weighted Evaluation (Divide Total Final Weighted Evaluation by 3; store it in Employee datatype array emp[num of emp].comp\_e\_evals.average)

End of TAFWE definition

Define SRinPercent function

Calculate Salary Raise in Percent (emp[num of emp].salraise) (using if statements to test value of Average Final Weighted Evaluation (emp[num of emp].comp\_e\_evals.average))

End of SRinPercent definition

Define Salary function

Calculate Salary raise in dollars (multiplying salary raise in percent (emp[num of emp].salraise) by 0.01 and then multiplying that by current salary (emp[num of emp].salary); store it in emp[num of emp].salraiseindoll)

Calculate Salary with raise (adding current salary (emp[num of emp].salary) to Salary raise in dollars (emp[num of emp].salraiseindoll); store it in emp[num of emp].newsal)

End of Salary definition

Define letter function

Determine if it is necessary to write a warning or congrats message to output file (using if statements; test Average Final Weighted Evaluation (emp[num of emp].comp\_e\_evals.average) value.

End of letter definition

Define report function

Write header (emp[num of emp].header[]) to output file

Write “Name of the Employee:” emp[num of emp].e\_name to output file

Write “Name of the Supervisor:” emp[num of emp].s\_name to output file

Write “Employee ID:” emp[num of emp].­­­­­­­e\_id to output file

Write “Telephone Number:” emp[num of emp].e\_tele to output file

Write “Address:” emp[num of emp].­­­­­­­­­address.street, emp[num of emp].­­­­­­­­­address.city, emp[num of emp].­­­­­­­­­address.state, emp[num of emp].­­­­­­­­­address.zip to output file

Begin evaluation output loop

Test year using if statements to print correct evaluations for each year

Write “Spring/Summer/Fall Semester Evaluation, XXXX” (XXXX is year) emp[num of emp].e\_evals.eval[semester num] to output file

End of evaluation output loop

Write “Final Weighted Evaluation, 2011:” emp[num of emp].comp\_e\_evals.compeval[0] to output file

Write “Final Weighted Evaluation, 2012:” emp[num of emp].comp\_e\_evals.compeval[1] to output file

Write “Final Weighted Evaluation, 2013:” emp[num of emp].comp\_e\_evals.compeval[2] to output file

Write “Total Final Weighted Evaluation:” emp[num of emp].comp\_e\_evals.total to output file

Write “Average Final Weighted Evaluation:” emp[num of emp].comp\_e\_evals.average to output file

Write “Current Salary:” emp[num of emp].salary to output file

Write “Salary Raise in %:” emp[num of emp].salraise to output file

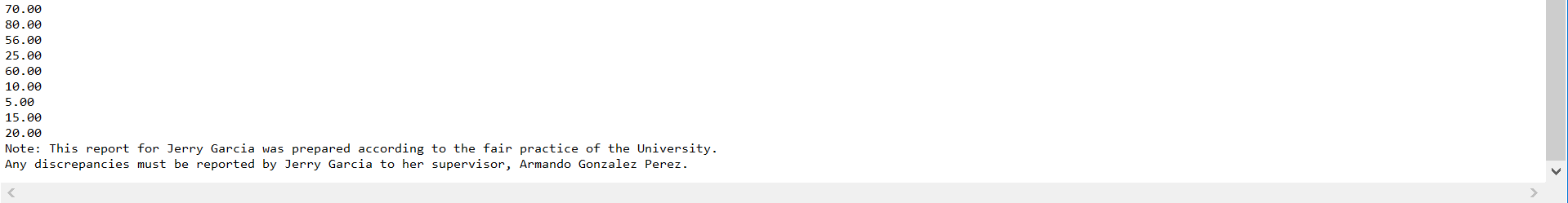
Write “Salary Raise in Dollars:” emp[num of emp].salraiseindoll to output file

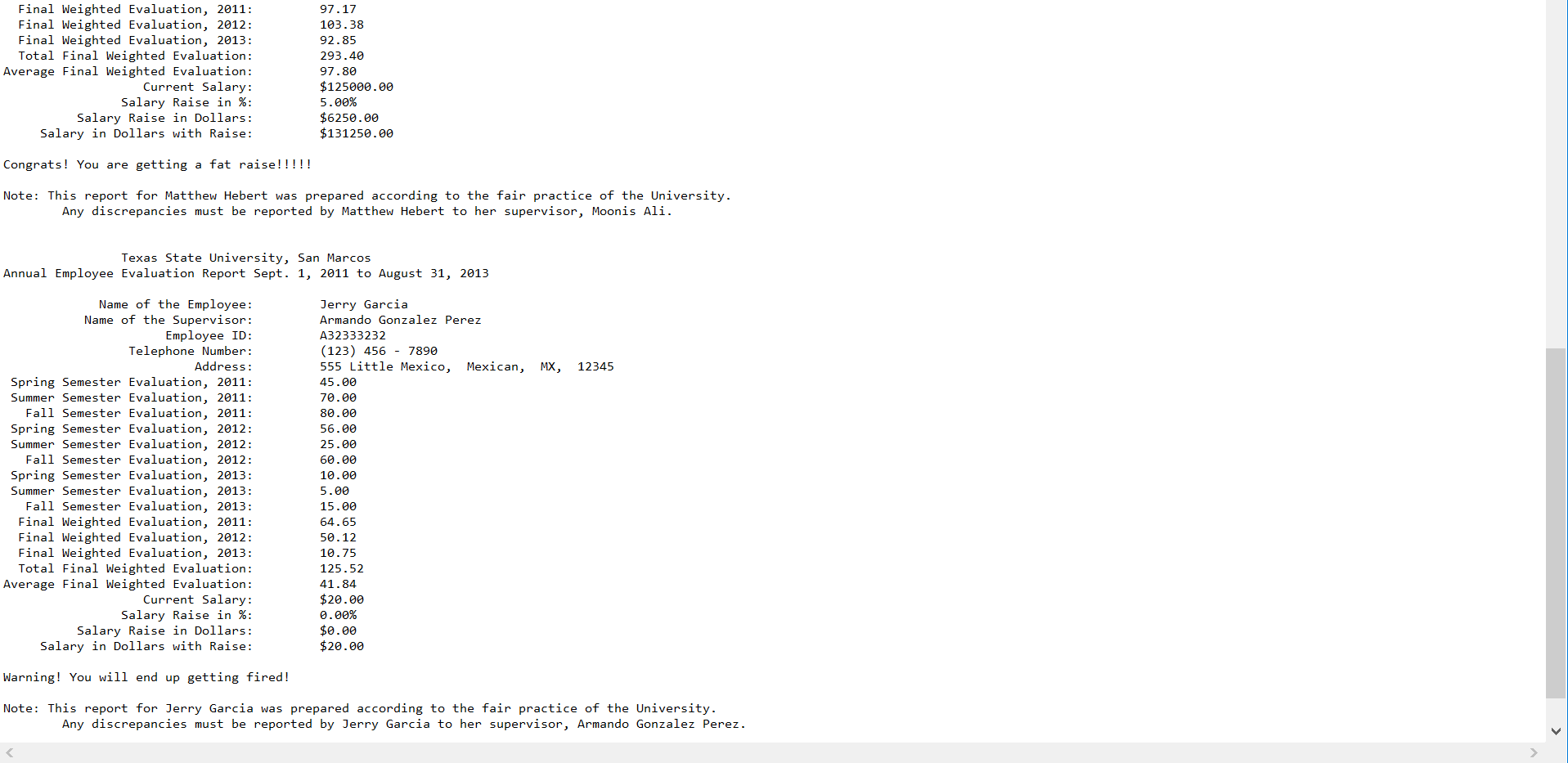
Write “Salary in Dollars with Raise:” emp[num of emp].newsal to output file

Call letter function (write warning/congrats message if needed)

Write discrepancy note (emp[num of emp].note[]) to output file

End of report definition

Input:

Output: