Programming Case Study

ICT30220 - Certificate III in Information Technology
ICTPRG302 - Apply introductory programming techniques

Case Study – “Work From Home Tracker“

Your role: You are a Programmer working for Phoenix IT Solutions.

Scenario

The current climate has forced many employees to work from home. Diamond Realty, a local real estate agent, has contacted you to write a program to help them track the hours worked by their seven (7) employees from home. The “Work From Home Tracker” program will track the daily hours worked from Monday to Friday, and then calculate the total weekly hours worked for each employee.

You have met with your client (teacher) and have obtained the project specifications for the program you will build as outlined below.

Program Specifications

Program Interface

The program needs to present a suitable interface (menu/form/web app depending on your chosen programming language) where staff can:

1. Enter Daily Hours Worked
2. Produce Hours Worked Report
3. Quit/Close/Exit the Program

Program Functionality

a) For Option **[1] Enter Daily Hours Worked**, your program must request and process data for all seven (7) employees. For each employee record, your program must ask for:

- The Current Working Week Number
- Employee ID
- Employee Name
- The hours worked from home for the 5 working days Monday – Friday. For this, create a data structure, **Data Structure 1**, such as a single dimensional array (or similar) to store the hours worked for each day of the week. For example:

Data Structure 1 – This stores the hours worked each day by an employee

Monday	Tuesday	Wednesday	Thursday	Friday

b) Once all hours have been entered, process the daily hours worked in **Data Structure 1** by outputting an appropriate message to the screen as shown in the table below (messages may be customised):

Hours worked	Error message output to screen
Less than 4 hours a day	Insufficient hours worked on <this day>"
More than 10 hours a day	Too many hours worked on <this day>
Less than 30 hours a week	You didn't do enough work this week
More than 40 hours a week	You are working too hard!!

For example, if an employee worked less than 4 hours on Thursday, the message "Insufficient hours worked on Thursday" will be output to the screen.

c) Your program should then write each employee record to file and include the following information:

- Week Number
- Employee ID
- Employee Name
- Hours worked for each day

A sample entry in your text file (or .csv file) might be:

```
Week 2, 123, Vicki Brainsworth, 3,6,8,12,9
Week 2, 345, Joey Genius, 8,12,6,11,7
```

d) Add up the daily hours worked for the employee and store this in a second data structure (such as a list), **Data Structure 2**. This information will be used later to produce the **Employee Weekly Report** on the screen after you have entered the work pattern for all employees. For example:

Data Structure 2 – This stores the total weekly hours worked for each employee

Employee 1	Employee 2	Employee 3	Employee 4	Employee 5	Employee 6	Employee 7
38	44	33	43	36	37	38

For example, Employee 5 worked 36 hours for the current week

e) Once all 7 (seven) employee weekly hours have been stored, process **Data Structure 2** and output the **Weekly Employee Report** to screen which displays:

- The number of employees who worked less than 30 hours a week
- The number of employees who worked more than 40 hours a week
- The number of employees who worked between 37-39 hours.

f) Once all records are processed, your Program must return to the Main Screen somehow.

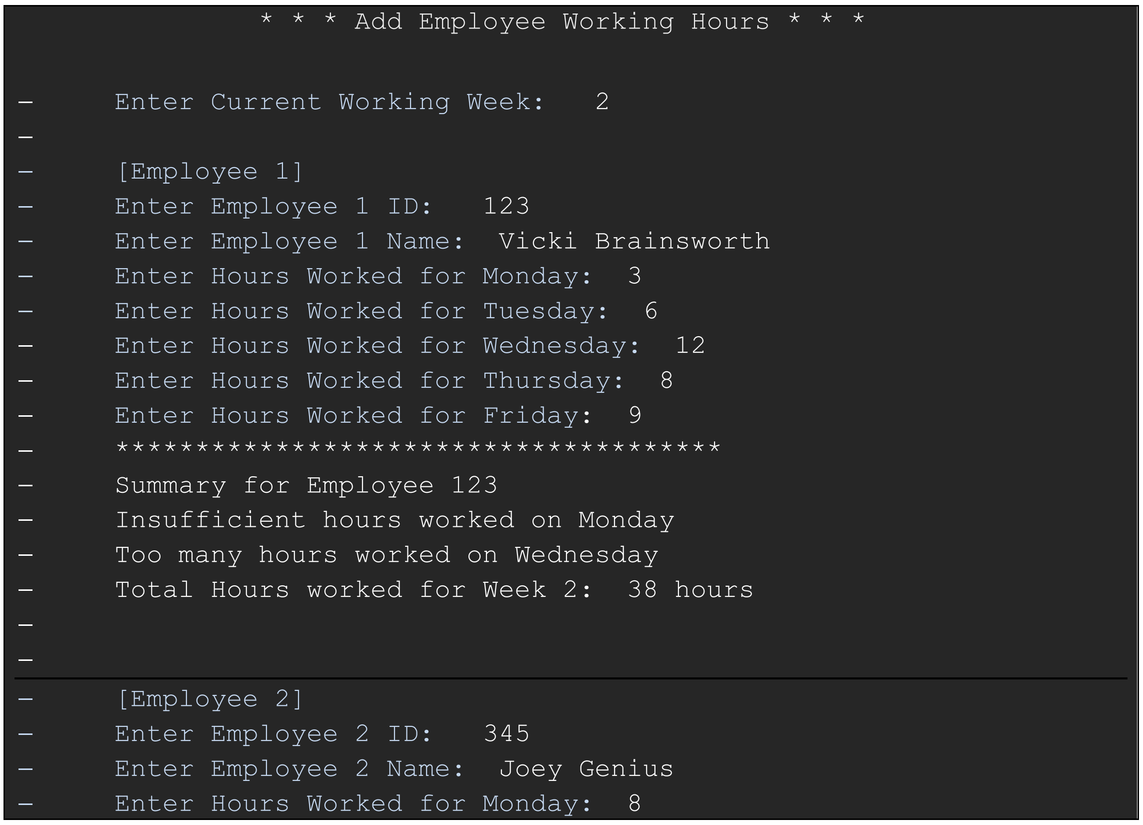
g) For Option **[2] Produce Hours Worked Report**, your program is required to read the employee records from file and display them to the screen with the latest entries at the top of the list (sorted). You must give the user the choice for the number of records to be displayed. For example, if the user enters “5”, then the 5 most recent records will be displayed. The program must be able to return back to the Main Screen.

h) For **Option [3]** your program must exit/close appropriately

Sample of “Work From Home Tracker” running as a Console Application

This sample shows what the program might look like running as a console application. This could vary depending on your interface (for example Web App, or Form with buttons etc). It is the functionality that is important.

This assumes the user has selected **[1] Enter Daily Hours Worked** from the Main Screen.



```
                * * * Add Employee Working Hours * * *

-     Enter Current Working Week:   2
-
-     [Employee 1]
-     Enter Employee 1 ID:   123
-     Enter Employee 1 Name:  Vicki Brainsworth
-     Enter Hours Worked for Monday:  3
-     Enter Hours Worked for Tuesday:  6
-     Enter Hours Worked for Wednesday:  12
-     Enter Hours Worked for Thursday:  8
-     Enter Hours Worked for Friday:  9
-     *************************************
-     Summary for Employee 123
-     Insufficient hours worked on Monday
-     Too many hours worked on Wednesday
-     Total Hours worked for Week 2:  38 hours
-
-
-     [Employee 2]
-     Enter Employee 2 ID:   345
-     Enter Employee 2 Name:  Joey Genius
-     Enter Hours Worked for Monday:  8
```

```
–      Enter Hours Worked for Tuesday:  12
–      Enter Hours Worked for Wednesday:  6
–      Enter Hours Worked for Thursday:  11
–      Enter Hours Worked for Friday:  7
–      ***************************************
–      Summary for Employee 345
–      Too many hours worked on Tuesday
–      Too many hours worked on Thursday
–      Total Hours worked for Week 2:  44 hours
–      You are working too hard!
–
–      . <continue adding records for all 7 employees>
–      . <once complete, a Weekly Employee Report should>
–      . <be displayed as shown below>
–
–      ***************************************************************
–                   Weekly Employee Report
–      Number of Employees who worked Less than 30 hours this week:  2
–      Number of Employees who worked more than 40 hours this week:  3
–      Number of Employees who worked Between 37-39 hours this week:
2
–
–      <Press 1 to return to the Main Screen or 2 to Exit > _
–
```

Your Task

Following basic language syntax rules (and relevant programming standards), you will develop the application for your client whilst following the **Mandatory Coding Specifications Checklist** below.

Mandatory Coding Specifications Checklist	
1. Program must make use of Sequence, selection and iteration constructs	☐
2. Usage of datatypes, operators, expressions	☐
3. Appropriate usage of operators and expressions	☐
4. Declare and use variables, appropriate data types, and variable scope	☐
5. Make use of at least two (2) library functions	☐
6. Usage of at least two (2) types of commenting techniques	☐
7. Expressions in selection and iteration using logical operators	☐
8. String manipulation techniques	☐
9. Usage of two (2) different Data Structures	☐
10. Reading and writing to a text file	☐

Client Requirements Checklist

Use the Client Requirements Checklist below to ensure application meets initial client specifications.

Client requirement checklist	**Completed (tick box if yes)**
Appropriate Interface and Layout suitable for application	☐
Application is suitable for target audience	☐
Application runs as required:	
a) Program allows user to enter all Employee Records	☐
b) Totals are calculated and displayed correctly	☐
c) Appropriate messages displayed to screen	☐
d) Records written to file correctly	☐
e) Records display correctly on the screen	☐
f) Program returns to Main Screen appropriately	☐
g) Program exits/closes appropriately	☐
Evidence of testing has been demonstrated to client	☐
Major bugs/errors have been rectified	☐
Game/Application is free from grammatical/spelling errors	☐

End of Case Study