

## CS21 Assignment #8

**Please use the given name for your scripts!**

1. (*payday.py*) Write a program to collect user input, compute total pay, and write user and summary information to a text file. Place all statements in `main()`.

### Step 1: Populate three (parallel) lists

Using the appropriate loop collect name (string), hours worked (float) and hourly rate (float) for employees. If all three inputs are valid place the three values, each in its own list. You will stop collecting the names once the user gives a blank name (hits enter)

- Prompt each user for their name, hours worked (float) and hourly rate (float). Append each value to its own list. You will not know in advance how many users to expect, so use the appropriate loop.
- Handle `ValueError` exceptions. If this error occurs, ask user to re-enter data. See Sample Run for example.

### Step 2: Create *payroll.txt*

- Using a loop, step through lists and compute each person's total pay. You do not need to store total pay in a list.
- Write each person's information to *payroll.txt* (see below). You may use '`\t`' in your write statement.
- Keep a running total. Write the total payroll and average paycheck to *payroll.txt*.
- Remember to close the file.

#### Sample Run for Step 1:

```
Name (just hit Enter when done): Singh
Hours worked: 40
Hourly rate: 15.75
Name (just hit Enter when done): Mustafa
Hours worked: 30
Hourly rate: 16.10
Name (just hit Enter when done): J
Hours worked: r5
Values must be numeric.
Please input employee's name and values again.
Name (just hit Enter when done): J
Hours worked: 10.2
Hourly rate: 15.25
Name (just hit Enter when done):
>>>
```

#### Sample Contents of *payroll.txt*

Singh	40.00	15.75	630.00
Mustafa	30.00	16.10	483.00
J	10.20	15.25	155.55
Total Payroll = \$1268.55			
Average Paycheck = \$422.85			

2. (*cities.py*) Write a program that asks user for a city name, compares that value with the contents of two text files, and reports each match (if any). Define a function to handle importing file data into a list.

### Step 1: Define function

Function	Input Argument	Processing	Output
file2list	file name	Open file for reading. <i>See textbook pages 322-3 (lines 5-11)</i> Read each city name and store in list. <i>See textbook pages 322-3 (lines 13-17)</i> Handle IOError exception by printing an error message and returning an empty list. <i>Suggestion: Use try-except-else construct.</i> Remember to close the file.	A list populated with the city names for that state

<b>Code you can use to test function:</b> <pre>mylist=file2list('vt_municipalities.txt') print('Your VT list contains',len(mylist),'elements.') mylist=file2list('nh_munic.txt') print('Your nh list contains',       len(mylist),'elements.')</pre>	<b>Result of test (NH file name spelled wrong):</b> Your VT list contains 123 elements. ***FILE ERROR: nh_munic.txt cannot be found. Your NH list contains 0 elements.
---	---

### Step 2: Use main() for user interaction

- Populate two lists, one each with the contents of *vt\_municipalities.txt* and *nh\_municipalities.txt*. To do this, call your `file2list` function.
- Use `len` function to be sure your lists are not empty.
- If either list is empty, do nothing. Otherwise, prompt the user for a city name and report the results as shown. Continuing prompting for additional city names until user enters 'q' for quit.

Sample Run:

```
City name (type 'q' to quit): Albany
Albany is in Vermont and New Hampshire.

City name (type 'q' to quit): Bethel
Bethel is in Vermont.

City name (type 'q' to quit): Bath
Bath is in New Hampshire.

City name (type 'q' to quit): Xanadu
Neither VT nor NH has a city by that name.

City name (type 'q' to quit): q
>>>
```

It is expected that you will use exception handling and error handling to ensure your programs are robust and will not crash.

**Reminders (not following will result in point deductions):**

If a function's task doesn't include output to the user, do not do it!

All programs should have a `main()` function

Use constants! No magic numbers! No global variables!

It is expected that you will complete the same process of development that we use in class. When you reach the point of having an algorithm (pseudocode), this will become the comments of your program as a starting point for writing code. Comment first, then code!

Be sure to include comments at the top of the program that include your name, class and a short description of the program.

Each function should begin with a comment describing the task the function will perform.

Be sure all output is formatted. Unless otherwise, specified, displays non-integer values with 2 digits after the decimal point.

*Any work you submit for this assignment should be authored entirely by yourself. Assistance is permitted from the instructor or teaching assistants only. All submitted programming assignments are subject to originality verification through software designed and used for the Measure Of Software Similarity (MOSS).*