Project 5: Payroll (Part 2)

CS 1410

Background

Look at the spec in Part 1 for background information.

Requirements (Part 2)

Implement your design From Part 1, including the five module-level functions used in the **main** function above by the final due date. Use *p5.py* as your main module.

After running your program, compare the three new entries in *paylog.txt* to *paylog_old.txt* that was copied earlier in **main**, and verify that the pay amount and method have been changed appropriately for the three employees updated in **main**. Submit your *payroll.py*, *paylog_old.txt*, and *paylog.txt* output files.

Implementation Notes

Every time **run_payroll** executes, you will first delete the previous *payroll.txt* file, if it exists (the main program in *p5.py* does this for you). Here is **run_payroll** (you can just use it):

```
def run_payroll():
    if os.path.exists(PAY_LOGFILE): # pay_log_file is a global variable holding ‘payroll.txt’
        os.remove(PAY_LOGFILE)
    for emp in employees:                # employees is the global list of Employee objects
        emp.issue_payment()              # issue_payment calls a method in the classification
                                         # object to compute the pay, which in turn invokes
                                         # the pay method.
```

Every time you issue the payment for an Hourly or Commissioned employee, clear their respective timecard or receipt list afterward, so these entries won’t be used again for the next pay period.

Remember that the concrete payment class (**MailMethod** or **DirectMethod**) is responsible for “delivering” the payment by writing to the log file. Therefore, you will want to open the pay log file in “append mode” using ‘a’ in the second argument to your call to **open** before writing to that file. Make sure the file is closed after appending each new payroll entry.

After you have run the main module in the file p5.py (in Canvas), *payroll.txt* should contain:

```
Mailing 5599.44 to Issie Scholard at 11 Texas Court Columbia Missouri 65218
Mailing 2584.64 to Reynard Lorenzin at 3233 Spaight Point Houston Texas 77030
Mailing 1880.00 to Jed Netti at 85 Coolidge Terrace San Antonio Texas 78255
```

The entries above indicate the format you should use for printing payroll results to *payroll.txt*. Use the **round** built-in function to achieve 2 decimals.

Here is a suggested development sequence:

1. Write **load_employees**. It opens *employees.csv*, ignores the first line, and then reads a line at a time, splitting its arguments on a comma. Create a new Employee object initialized with the

string attributes. Then create the appropriate instances for the employee’s classification and add it as an attribute to the new Employee object. Finally, add the Employee object to your global list of employees.

2. Write **find_employee_by_id** by searching the list of employees and returning the Employee object.
3. Implement **Employee**
4. Implement the **Classification** Hierarchy
5. Implement **process_timecards**
6. Implement **process_receipts**