Project 5: Payroll (Part 2)

CS 1410

Background

Look at the spec in Part 1 for background information.

Requirements

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, 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 timecard or receipt lists, respectively, 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 1911.87 to Karina Gay at 998 Vitae St. Atlanta GA 45169
Transferred 2547.52 for TaShya D. Snow to 465794-3611 at 30417353-K
Mailing 861.20 to Rooney Alvarado at 4963 Nisl. St. Ap #185 Gillette WY 20226
```

Here is a reasonable development sequence:

1. Write **load_employees**. It opens *employees.csv*, ignores the first line, and then read 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 payment method and bind them as attributes 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**
7. Implement the **PayMethod** hierarchy