The University of Texas
Computer Science

313E Programming Assignment 04
Employee Salaries

Setup

This assignment may be completed individually or with a pair programming partner. Be sure to include your and your partner's name and EID in the Python file header. If you are working alone, you may delete one of them.

Complete these steps to prepare for the Assignment.

Check	Description
▢	Download employee.py and employee.out. You will be working on the employee.py file.
▢	Place both files in the same folder/directory.
▢	You may not change the file names. Otherwise, the grading script will not work.

employee.out is a sample of what your output should look like when your program is complete. You do not have to edit this file.

If you would like to turn on autosave in VSCode, click on File -> Autosave.

Problem Description

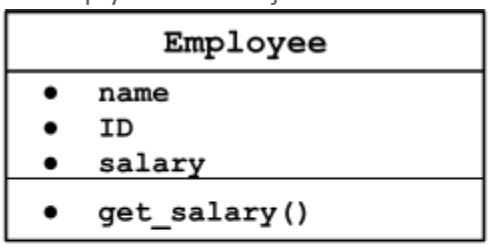
Requirements

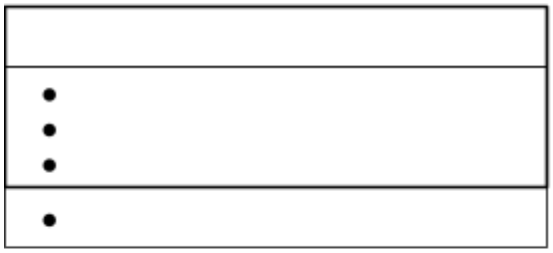
A company has different types of employees and salaries are calculated differently for several employee types. For this programming assignment, you will complete the methods for each class, each class being a different type of employee. The employee types are described in the following table.

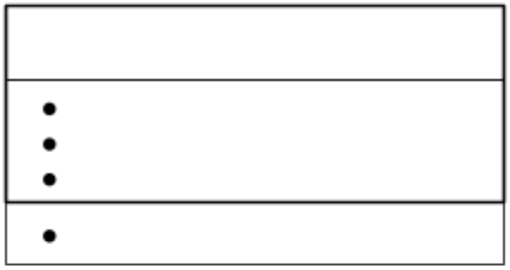
Type	Description
Employee	All employee types are also an Employee. Employees have a name, ID and salary.
Permanent Employee	Permanent employees can choose to have the following benefits: health insurance and/or retirement. The salary value is annual. if benefits = health insurance then salary = base_salary * 0.9 if benefits = retired, then salary = base_salary * 0.8

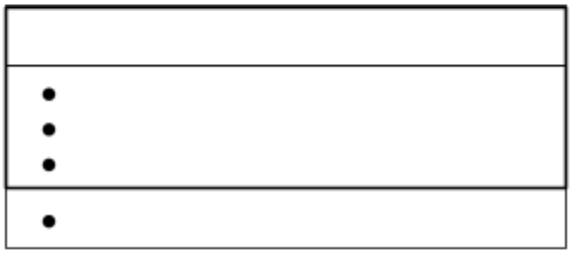
	If benefits = health insurance and retired, then salary = base_salary * 0.7
Temporary Employee	Temporary Employees have hours, the number of hours they work per month. The salary value is hourly. salary = base_salary * hours
Consultant	Consultants are a type of Temporary Employee. Consultants have trips, representing the number of trips required per month. Consultants get paid $1000 for each trip they take. salary = (base_salary * hours) + (trips * 1000)
Manager	Managers have a bonus representing the monthly bonus amount. salary = base_salary + bonus
Consultant Manager	Consultant Managers are a type of Consultant and a type of Manager. salary = (base_salary * hours) + (trips * 1000) + bonus

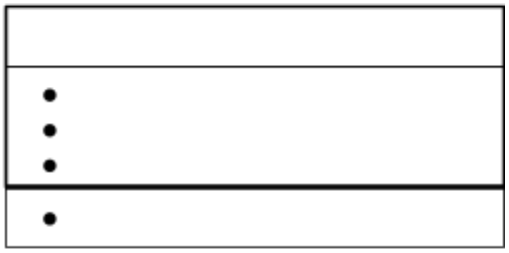
To help you get started, you can add the class, field, and method names to the class design below. The Employee class has been done for you. Try adding an arrow from the superclass to the subclass to help visualize the inheritance relationship and help you write object-oriented code.

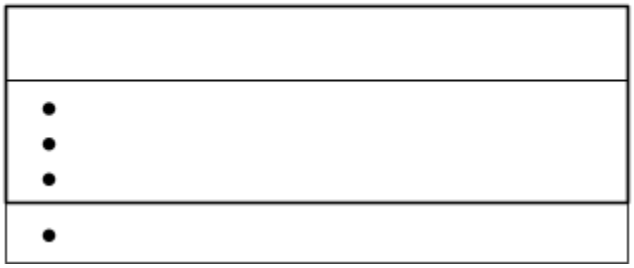












This assignment includes the following technical requirements:

1. **Inheritance must be used appropriately.**
2. **Each class should have a string method implementation that returns a string representation of the object.**
3. **Each class should have the necessary getter and setter methods for attributes.**
4. **The appropriate classes should have a method that calculates the salary for that kind of employee.**

Input

No input file is needed for this program. The starter code in main() will provide all the needed data.

Output

Code to produce the output already exists in main(). The existing code will call the methods that you wrote in your employee classes. Compare the output of your code with the sample output file to verify that you have the correct output.

Submission

Follow these steps for submission.

Check	Description
▢	Verify that you have no debugging statements left in your code.
▢	Submit **only** employee.py (the starter code file with your updates) to the given assignment in Gradescope. This will cause the grading scripts to run.
▢	If you have a partner, make sure you are submitting it for you **and** your partner on Gradescope. See this Gradescope documentation for more details.
▢	When the grading scripts are complete, check the results. If there are errors, evaluate the script feedback, work on the fix, test the fix, and then re-submit the file to Gradescope. You can submit as many times as necessary before the due date. NOTE: By default, only the most recent submission will be considered for grading. If you want to use a previous submission for your final grade, you must activate it from your submission history before the due date.

Academic Integrity

Please review the Academic Integrity section of the syllabus. We will be using plagiarism checkers, and we will cross-reference your solution with AI-generated solutions to check for similarities. Remember the goal in this class is not to write the perfect solution; we already have many of those! The goal is to learn how to problem solve, so:

- don't hesitate to ask for guidance from the instructional staff.
- discuss with peers about **only** bugs, their potential fixes, **high-level** design decisions, and project clarifying questions. Do not look at, verbalize, or copy another student's code when doing so.

Attribution

Thanks to Dr. Kia Teymourian for the assignment requirements and Dr. Carol Ramsey for the instruction template.