CSE 241 Homework Assignment 6

DUE

May 12, 2019, 23:55

Description

• This is an individual assignment. Please do not collaborate.

• If you think that this document does not clearly describe the assignment, ask questions before it's too late.

You won't be given a chance to correct any mistakes.

Scope: University Management System

You are expected to develop university management system.

Main Problem: Personal Management

This HW is extended version of HW5. That is to say, this homework is also storyteller, but it tells story of YOK and multiple universities. Actors of the story are employees. However, their job is not specified. It is determined by YOK according to applied education system. Also, universities are also founded by YOK. So, YOK has got two responsibilities; employ new personnel and found new university.

Employee is an abstract class which has got a virtual function; work.

Your Program is storyteller. Actors of the story are 10 employees who work in the University. It tells a story which composes 50 random actions. Actions are described before. Actions can be taken by only actor stated below. Actors can be Lecturer, ResearchAssistant, Secretary or Officer, described in UML diagram. They are inherited from Employee. As you see, some actions can be taken by all employee types.

YOK has 2 functions;

- Employ(name,surname,type):Employee
  İt takes name and surname of the person and it construct new employee
- buildUniversity(type):University
  if research uni -> produce Univeristy<double>
  if linguistic uni -> produce University<int>
  if technical uni -> produce University<Complex>

where comlex is new data type which contain both int and double, overloading +, ++, -, = operators.

Requirements:

1. Your program will not expect any input. When it begins, it will process and prints all story.
2. Program should read txt file, personnellist.txt. It indicates Employee. There are 30 personnel examples in list. Universities employ them by giving job.
3. There is a University<T> data type in program. It employs Employee and it expects that personnel contributes.
4. University has a template for contribution.
5. Actions should be managed with enumeration.
6. You should separate compilation into header files and implementation files.
7. Acions are handled by a virtual function work. It takes an action as input. It is implemented by all job types in the way they do their job. Other actions which the employee can not do will be handled by error handling. There will be an exception type to handle this situation.

Test Scenarios

Firstly, program reads employeelist.txt. Your program founds 3 universities. University names are given by you. This is also a part of story. Then, Universities employ 10 employees. In other mean, 10 employees are selected randomly from the list and 3 universities employs them through YOK. Then, 50 random actions are processed for each university, and these actions will be taken by related employee, which stated in the list. These actions affect happiness of employee and contribution of university. Work function of classes tell the story(print which action happens) and do effects. Actions are produced inside of the universities. Each university produce its action.

University requires an employee. YOK give job to a person. University employ him/her. YOK factorize university and employee. There will be an enumeration about employee type and university type, and YOK produce new employee and university according to types taken as input.

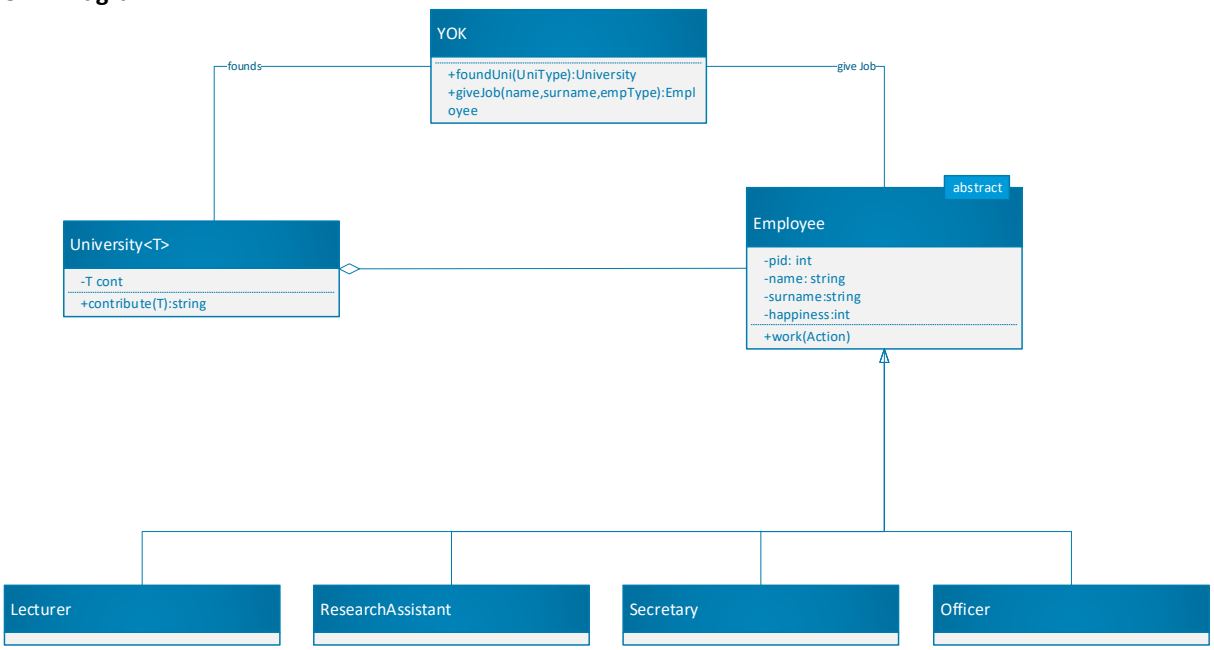
For example;

Suppose that University (GTU) needs to employ Lecturer, ResearchAssistant. It requests from YOK. YOK gives these jobs to some people indicated in the list. Let's say these are Ahmet Dizdar and Furkan Bozgun. First action is slackness. Then Program selects one employee, let us say that it is Ahmet Dizdar, working in the university and it takes the action by doing function related. This function, which is drinkTea, prints the action and do effects, increasing the happiness of the actor and contribution of the university. So, happiness of A is 5 and contribution of uni is -2. Let Second action be administration. Then program selects one AdministrativePersonnel, let us say that it is D, working in the university and it takes the action by doing

function related. This function, which is manageProcess, prints the action and do effects, increasing the happiness of the actor and contribution of the university. So, happiness of D is -1 and contribution of the uni is +2.
So, it prints like that;
(output)> GTU requests Lecturer
(output)>YOK give job to Ahmet Dizdar as lecturer.
(output)> GTU employs Ahmet Dizdar as Lecturer.
(output)> GTU requests Officer
(output> YOK give job to Furkan Bozgun as Officer
(output)>GTU employs Furkan Bozgun as Officer.
(Output)> Ahmet Dizdar have slackness. Therefore, Ahmet Dizdar drinks tea. Happiness of Ahmet Dizdar is 5, contribution of uni is -2.
(Output)> Furkan Bozgun have administration. Therefore, Furkan Bozgun manage process. Happiness of Furkan Bozgun is -1, contribution of uni is 0.

As you see, contribution of university is common, i.e. contribution of university is affected by all action. But happiness of actor is affected by only the action the actor takes. After finishing all actions, happiness of all actors and contribution of university will be shown.

UML Diagram



Remarks

- Do not use any elements which is not covered in class.
- Do not submit your code without testing it with several different scenarios.
- Write comments in your code.
- Prepare a Makefile for your programme.
- You are free to define your functions or classes.

Turn in:

- Source code of a complete C++ program. Name of the file should be in this format: <full_name>_<id>.cpp.
- Example: gokhan_kaya_000000.cpp. Please do not use any Turkish special characters.
- You don't need to use an IDE for this assignment. Your code will be compiled and run in a command window.
- Your code will be compiled and tested on a Linux machine(Ubuntu). GCC will be used
- Make sure you don't get compile errors when you issue this command : g++ <full_name>_<id>.cpp.
- A script will be used in order to check the correctness of your results. So, be careful not to violate the expected output format.
- Provide comments unless you are not interested in partial credit. (If I cannot easily understand your design,

you may loose points.)
• You may not get full credit if your implementation contradicts with the statements in this document.

Late Submission

• (0,24] hours: -20%
• (24,48] hours: -40%
• (48,72] hours: -60%
• (72,-) hours: -100%

Grading (Tentative)

• Max Grade : 100.
• Multiple tests(at least 5) will be performed.
All of the followings are possible deductions from Max Grade.
• #define HARD_CODED_VALUES -10.
• No submission: -100. (be consistent in doing this and your overall grade will converge to N/A) (To be specific: if you miss 3 assignments you'll get N/A)
• Compile errors: -100.
• Irrelevant code: -100.
• Major parts are missing: -100.
• Unnecessarily long code: -30.
• Using language elements and libraries which are not allowed: -100.
• Not caring about the structure and efficiency: -30. (avoid using hard-coded values, avoid hard-to-follow expressions, avoid code repetition, avoid unnecessary loops).
• Significant number of compiler warnings: -10.
• Not commented enough: -5. (Comments are in English).
• Source code encoding is not UTF-8 and characters are not properly displayed: -5. (You can use 'Visual Studio Code', 'Sublime Text', 'Atom' etc. . . Check the character encoding of your text editor and set it to UTF-8).
• Missing or wrong output values: Fails the test.
• Output format is wrong: -30.
• Infinite loop: Fails the test.
• Segmentation fault: Fails the test.
• Fails 5 or more random tests: -100.
• Fails the test: deduction up to 20.
• Prints anything extra: -30.
• Unwanted chars and spaces in output.txt: -30.
• Submission includes files other than the expected: -10.
• Submission does not follow the file naming convention: -10.
• Sharing or inheriting code: -200