NTOU Java Programming Homework 2

SOSEL
Service-Oriented Software Engineering Lab

Spring 2024

Goals

- 建構子(Constructor)
- 抽象類別(Abstract Class)
- 介面(Interface)
- 多型(Polymorphism)

Homework 2-1₁

- Please design a salary calculation application, which includes classes as follows:
  - *Employee* is abstract class, with an abstract method*: int getEarnings ()*.
  - *Employee* has subclasses:
    - (1) *SalariedEmployee*, which receives weekly salary: *weeklySalary*weeks*.
    - (2) *HourlyEmployee*, which receives hourly salary.
      - If the working hours in a week are less than or equal to 40 hours, the salary is *hourlySalary*hours*.
      - If the working hours are more than 40 hours, the salary for the part exceeding 40 hours is multiplied by 1.5 times.
    - (3) *CommissionEmployee*, which receives salary: *commissionRate*grossSales.*
    - (4) PieceWorker, which receives salary: *wage*piece* (number of completed works).

Homework 2-1₂

- Bonus is an interface used to define additional bonus. There are two implementation classes: *StaticBonus* and *DynamicBonus* (as attached):
  - *StaticBonus*: sets the bonus fixed at 10,000.
  - *DynamicBonus:* sets the bonus as 10% of salary.
- *Employee* must contain a field whose type is *Bonus*, which can be assigned to a *StaticBonus* or *DynamicBonus* object to calculate the bonus (員工可以有兩種Bonus形式).

Homework 2-1₃

- According to the above description, please design another class *EmployeeDataCollector* as the user interface:
  - Providing a text-based interactive UI to allow users to
    - choose the type of employees that should be added (by the Scanner API),
    - enter the required information for the selected type (for example, if you select *SalariedEmployee*, you need to enter the weekly salary and the number of working weeks), and
    - enter the bonus type (static or dynamic).
  - If the user inputs “-1”, the program should terminate, and display all entered employee details, total salary, and total bonus.

Homework 2-1$_4$

- All subtypes of *Employee* objects (Employee的子類別物件) need to be stored in an array of type *Employee*.
- All Employee objects in the array should be processed:
  - *toString()* is called to print the details.
  - *getEarnings()* is called to obtain individual salary (for summing up and output the summed salaries)
  - *getBonus()* of the Bonus field is called to get the bonus salary (for summing up and output the summed bonuses)
- Please note that the following six types of variables cannot appear in the program: *SalariedEmployee, HourlyEmployee, CommissionEmployee, PieceWorker, StaticBonus, and DynamicBonus.*

Sample Output

- For expected results, please refer to hw-2-1-sampleOutput.txt.

Hints

- Please refer to the textbook examples 10.4~10.9.
- An abstract *inputData* method can be added to the class *Employee*, and different input tasks can be implemented in the subclasses.
- The Bonus object should be set as a field of the Employee class.
- Please note that each *Employee* object may include either *StaticBonus* or *DynamicBonus object* for the field *bonus.*

Homework 2-2[1]

- Please design an advanced version of the RPG game, including the following classes:
  - *WarTest.java* (functions cannot be deleted): entry point.
  - *ATK.java* (functions cannot be deleted): It is the interface of the "attackable" object. The *attack()* method should return the amount of damage.
  - *Weapon.java* (functions cannot be deleted): It is the abstract class of the weapon, which implements the ATK interface, and includes two fields: *offense* (attack damage) and *defense* (defense degree).
  - *Pet.java* (need to be expanded): It implements the ATK interface to construct a pet with attack capability. Its “attack” damage is a random number from 1 to *maxAttack*.

Homework 2-2$_2$

- Player.java (need to be expanded):
  - It includes two attributes: *hp* (health point) and *equipment*.
  - *Equipment* represents weapons or pets. Multiple weapons or pets can be equipped at the same time to cause damage to the enemy at the same time (the number of weapons or pets cannot be fixed, thus *ArrayList* must be used).
  - Please add a constructor: *public Player (String name, double hp)* to set the given *hp*, and set the *equipment* to only one weapon with attack value of DEFAULT_ATK and defense value of 0.
  - Please add a constructor: *public Player (String name)* to set *hp* to *DEFAULT_HP*, and set *equipment* to the same setting as the above constructor.

Homework 2-2₃

- Please add three concrete subclasses of *Weapon*:
  - *NormalWeapon.java*: After attacking, the damage to the enemy's *hp* is the *attack* value of this weapon.
  - *DoubleWeapon.java*: It has a 1/5 chance of achieving two attacks, that is, the enemy's damage has a 4/5 chance of being the weapon's attack value, and 1/5 is double the attack value.
  - *PowerWeapon.java*: It increases the attack power by 20% each time, that is, the enemy's damage is 1.2 times the weapon's attack value.

Homework 2-2$_4$

- Please continue to design *War.java* to be responsible for battle-related functions (The default battle parties are the Player and the NPC):
  - Please design *battle ()*: the actual battle system to allow users to choose: "1.攻擊 2.防禦 3.結束".
  - If you choose "攻擊", the NPC's damage is the sum of the returned values of *attack()* of all weapons or pets in Player's equipment, and then deduct the defense value of all the NPC's equipment. The NPC will continue to attack, and the effect is reversed (the NPC's attack value deducts the Player's defense power).
  - If you choose "防禦", the NPC will not receive any damage. Although the NPC will attack the Player, the Player's defense value will be doubled. Besides, the Player will have a 1/3 chance to trigger a "self-healing" event, and the *hp* will increase a random number of 10~150.

Homework 2-2₅

- If the *hp* of either Player or NPC is less or equal to 0, the message “玩家被NPC擊倒了！” or “玩家擊倒了NPC！” will be displayed, and the game will end.
- According to the above description, please design the following three methods:
  - *public void attack(Player player1, Player player2, double defenseRate)*
  - *public void selfHeal(Player player)*
  - *public boolean determineVictory (Player user, Player npc)*
- Please note that the following variables cannot appear in the program:
  - *NormalWeapon, DoubleWeapon, PowerWeapon, Pet*

Sample Output

- Please refer to hw-2-2-sampleOutput-1.txt and hw-2-2-sampleOutput-2.txt

Hint

- This homework requires a lot of random numbers, please check the usage.
- You need to be familiar with the polymorphism concept.
  - When attacking, you need to call the *attack()* method of all ATK-typed objects at once.

Requirements

- The naming should conform to the CamelCase style.
- "Package" is required: ntou.cs.java2024.
- There must be comments in the class (at least your student number and name, and brief descriptions for this class and each method).
- Each assignment must have more than two classes, one of which is a test class (only including main).
- Please submit files including .java files and .class files (upload them to TronClass).
- Code that fails to compile or execute is not accepted.