Program 2: 50 points: Polymorphism/Abstraction Due: Thursday, October 1st, 11:59pm

**Objectives: The focus of this assignment is modifying and utilizing the inheritance hierarchy from the previous assignment. You will need to understand and implement the ideas of basic inheritance, polymorphism, and abstraction.**

**Program Description:**

Building upon the examples shown in class, this project will be a basic record of employees for a company. Many options for changing the list as a whole and updating the individual elements will be provided by the project.

A total of six classes are required. The first four are the same from the previous assignment with potential minor changes.

**Employee** (Altered from previous assignment)

**HourlyEmployee** (From previous assignment)

**SalaryEmployee** (From previous assignment)

**CommissionEmployee** (From previous assignment)

**EmployeeManager** – A class that contains an array of Employees and provides utilities to manage the array and the Employees stored within

**EmployeeDriver** – Contains main method. Creates a single EmployeeManager as well as the menu system in order to give the user the ability to use it. This is provided to you. DO NOT MAKE CHANGES

**UML DIAGRAM FOR AND DISCUSSION FOR Employee**

|  |
| --- |
| Employee **{abstract}** |
| - String firstName  - String lastName  - char middleInitial  - boolean fulltime  - char gender  - int employeeNum |
| <<constructor>> Employee (fn : String, ln : String, m : char, g : char, empNum : int, ft : boolean )  + getEmployeeNumber() : int  + setEmployeeNumber(empNum : int)  + getFirstName() : String  + getLastName() : String  + setFirstName(fn: String)  + setLastName(ln : String)  + setMiddleI(m : char)  + setGender(g : char)  + equals(e2 : Object) : Boolean  + toString() : String  **+ calculateWeeklyPay() : double {abstract}**  **+ annualRaise() {abstract}**  **+ holidayBonus() : double {abstract}**  **+ resetWeek() {abstract}** |

Notes on Data Members

The employeeNum member must be between 10000 and 99999, inclusive. If an invalid value is passed, the Employee class should immediately ask for another number until an acceptable one is given. This should be handled in setEmployeeNumber.

If an invalid value for gender is given (not ‘M’ or ‘F’) it should default to ‘F’. This should be handled in setGender.

Notes on Methods

equals() – Overrides Object equals(). Returns true if the employeeNum of the two instances are equal, false otherwise.

toString() – Overrides Object toString(). Returns as String of the Employee in the following format:

12345

Doe, John M.

Gender: M

Status: Full Time

(Note that Status doesn’t say true or false, rather Full Time or Part Time)

calculateWeeklyPay() – Abstract method to be implemented by subclass. Calculates pay for the week.

annualRaise() – Abstract method to be implemented by subclass. Gives Employee a raise.

holidayBonus() – Abstract method to be implemented by subclass. Calculates bonus for Employee.

resetWeek() – Abstract method to be implemented by subclass. Resets the weekly values for Employee.

**DISSCUSSION ON Employee SUBCLASSES**

**HourlyEmployee**

|  |
| --- |
| HourlyEmployee extends Employee |
| - double wage  - double hoursWorked |
| <<constructor>> HourlyEmployee (fn : String, ln : String, m : char, g : char, empNum : int, ft : boolean, wage : double )  + increaseHours(hours : double)  + toString() : String  + calculateWeeklyPay() : double  + annualRaise()  + holidayBonus() : double  + resetWeek() |

Additional Data Members:

* double wage
* double hoursWorked

Methods

Constructor accepts all that an Employee requires as well as a double for wage, hoursWorked set at 0.0.

Override toString(), returns a String of the HourlyEmployee in the following format:

12345

Doe, John M.

Gender: M

Status: Full Time

Wage: 3.40

Hours Worked: 0.00

*Abstract method implementation:*

calculateWeeklyPay() – Return amount earned in the week using wage and hoursWorked, any hours worked over 40 give double pay

annualRaise() – Wage is increased by 5%

holidayBonus() – Return amount of 40 hours worked (40\*wage)

resetWeek() – Resets hours worked to 0

increaseHours() - This class also needs the ability to increase the hours worked. Requesting to increase by a negative value should give no change, and report an error to the user.

**SalaryEmployee**

|  |
| --- |
| SalaryEmployee extends Employee |
| - salary : double |
| <<constructor>> SalaryEmployee (fn : String, ln : String, m : char, g : char, empNum : int, ft : boolean, salary : double )  + toString() : String  + calculateWeeklyPay() : double  + annualRaise()  + holidayBonus() : double  + resetWeek() |

Additional Data Members

* double salary

Methods

Constructor accepts all that an Employee requires as well as a double for salary.

Override toString(), returns a String of the SalaryEmployee in the following format:

12345

Doe, John M.

Gender: M

Status: Full Time

Salary: 50000.00

*Abstract method implementation:*

calculateWeeklyPay() – Return amount earned in the week by dividing salary by 52

annualRaise() – Salary is increased by 6%

holidayBonus() – Return 3% of salary

resetWeek() – No change

**CommissionEmployee**

|  |
| --- |
| CommissionEmployee extends Employee |
| - sales : double  - rate : double |
| <<constructor>> CommissionEmployee (fn : String, ln : String, m : char, g : char, empNum : int, ft : boolean, rate : double )  + increaseSales(sales : double)  + toString() : String  + calculateWeeklyPay() : double  + annualRaise()  + holidayBonus() : double  + resetWeek() |

Additional Data Members

* double sales
* double rate (stored as a percent, eg. 3.5% would be stored as 3.5)

Methods

Constructor accepts all that an Employee requires as well as a double for rate, sales set to 0.0.

Override toString(), returns a String of the CommissionEmployee in the following format:

12345

Doe, John M.

Gender: M

Status: Full Time

Rate: 3.50

Sales: 0.00

*Abstract method implementation:*

calculateWeeklyPay() – Return rate percentage of sales

annualRaise() – Rate percentage increased .2% *example, if rate was 2.5, it becomes 2.7*

holidayBonus() – No bonus

resetWeek() – Reset sales to 0.0

*Additional functionality:*

increaseSales() - This class also needs the ability to increase the sales. Requesting to increase by a negative value should give no change and report the error.

**EmployeeManager**

|  |
| --- |
| EmployeeManager |
| - employees : Employee[]  - employeeMax : final int = 10  -currentEmployees : int |
| <<constructor>> EmployeeManager  + addEmployee( type : int, fn : String, ln : String, m : char, g : char, en : int, ft : boolean, amount : double)  + removeEmployee( index : int)  + listAll()  + listHourly()  + listSalary()  + listCommision()  + resetWeek()  + calculatePayout() : double  + getIndex( empNum : int ) : int  + annualRaises()  + holidayBonuses() : double  + increaseHours( index : int, amount : double)  + increaseSales( index : int, amount : double) |

Data Members

* Employee[] employees – Collection of Employee objects
* final int employeeMax = 100 – Maximum number of Employees allowed
* int currentEmployees – Current number of Employees in collection

Methods

public EmployeeManager()

Constructor, creates the Employee array, sets currentEmployees to 0.

public void addEmployee(int, String, String, char, char, int, Boolean, double)

Takes an int representing the type of Employee to be added (1 – Hourly, 2 – Salary, 3 – Commission) as well as the required data to create that Employee. If one of these values is not passed output the line, “Invalid Employee Type, None Added”, and exit the method. If an Employee with the given Employee Number already exists do not add the Employee and output the line, “Duplicate Not Added”, and exit the method. If the array is at maximum capacity do not add the new Employee, and output the line, "Cannot add more Employees".

public void removeEmployee(int)

Removes an Employee located at the given index from the Employee array, the resulting order does not matter.

public void listAll()

Lists all the current Employees. Outputs there are none if there are none.

public void listHourly()

Lists all the current HourlyEmployees. Outputs there are none if there are none.

public void listSalary()

Lists all the current SalaryEmployees. Outputs there are none if there are none.

public void listCommission()

Lists all the current CommissionEmployees. Outputs there are none if there are none.

public void resetWeek()

Resets the week for all Employees.

public double calculatePayout()

Returns the total weekly payout for all Employees.

public int getIndex(int)

Given an Employee Number, returns the index of that Employee in the array, if the Employee doesn’t exist retuns -1.

public void annualRaises()

Applies annual raise to all current Employees.

public double holidayBonuses()

Outputs the holiday bonus for each Employee, and returns the total holiday bonus of all Employees.

public void increaseHours(int, double)

Increase the hours worked of the Employee at the given index by the given double amount.

public void increaseSales(int, double)

Increase the sales of the Employee at the given index by the given double amount.

Some notes:

* To begin with the EmployeeManager has no Employees
* There should never be two Employees with the same number, the newer Employee would be removed

**EmployeeDriver**

The EmployeeDriver will create a single object of EmployeeManager as well as any other variables you deem necessary. This is the interface to using the EmployeeManager through a Menu system. The Main Menu should look like this:

Main Menu

No Employees.

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Quit

Enter Choice:

Where “No Employees” is listed, **ALL** of the current Employees must be listed if there are any, “No Employees” if there are none. Some of the options lead to a submenu which will allow the user to return back to the previous menu. Whenever the main menu is displayed the employees will be listed as they were added. This is provided for you. You must use the version provided and cannot make any changes.

Menu Option Details

*1. Employee Submenu*

This leads the user to a submenu where they can choose a specific type of Employee. Once one is selected **ONLY THE EMPLOYEES OF THAT TYPE** are listed. Employees of type HourlyEmployee and CommissionEmployee are offered the option to add hours/sales as well as return to the previous menu. SalaryEmployees can only return to the previous menu.

Main Menu

12345

Parker, Peter B.

Gender: M

Status: Full Time

Wage: 8.50

Hours Worked: 0.00

54321

Spector, Marc J.

Gender: M

Status: Full Time

Wage: 9.80

Hours Worked: 0.00

98765

Pryde, Kitty T.

Gender: F

Status: Part Time

Salary: 25000.00

45678

Curry, Arthur J.

Gender: M

Status: Full Time

Salary: 65000.00

87654

Kent, Clark J.

Gender: M

Status: Part Time

Rate: 3.40

Sales: 0.00

1. Employee Submenu

2. Add Employee

3. Remove Employee

4. Calculate Weekly Payout

5. Calculate Bonus

6. Annual Raises

7. Reset Week

8. Quit

Enter Choice: 1

1. Hourly Employees

2. Salary Employees

3. Commission Employees

4. Back

Enter Choice: 1

12345

Parker, Peter B.

Gender: M

Status: Full Time

Wage: 8.50

Hours Worked: 0.00

54321

Spector, Marc J.

Gender: M

Status: Full Time

Wage: 9.80

Hours Worked: 0.00

1. Add Hours

2. Back

Enter Choice: 2

1. Hourly Employees

2. Salary Employees

3. Commission Employees

4. Back

Enter Choice:

*2. Add Employee*

Gives a similar submenu, asking which type of Employee to add, prompting for the appropriate information. Returns to main menu after added.

*3. Remove Employee*

Asks for an employee number and removes the Employee from the array if it exists. If not states that no such Employee exists and none have been removed. Returns to main menu after.

*4. Calculate Payout*

Reports how much the company will have to pay out given the Employee’s current sales, hours worked, and salaries

*5. Calculate Bonus*

Lists all Employees as well as their specific holiday bonus amount after their information. Also reports the total holiday bonus payout.

Enter Choice: 6

12345

Parker, Peter B.

Gender: M

Status: Full Time

Wage: 7.80

Hours Worked: 0.00

Bonus Amount: 312.00

54321

Wayne, Bruce J.

Gender: M

Status: Full Time

Wage: 9.50

Hours Worked: 0.00

Bonus Amount: 380.00

98765

Richards, Reed L.

Gender: M

Status: Part Time

Salary: 20000.00

Bonus Amount: 600.00

65432

Prince, Diana R.

Gender: F

Status: Full Time

Rate: 55000.00

Sales: 0.00

Bonus Amount: 0.00

Total holiday bonus payout is 1292.00

*6. Annual Raises*

Applies the annual raises for all Employees and notifies that annual raises have been applied.

*7. Reset Week*

Resets the weekly values for all Employees.

*8. Quit*

Exits the program giving an exiting message.

**Other Notes:**

* Make sure monetary values are formatted correctly
* Monetary amounts are forced to be 2 decimal places
  + Keep this in mind when applying raises, for example if a wage is 9.50, increasing by 5% gives 9.975, this should be rounded DOWN to 9.97
* Make proper use of super
* When down casting to program in the specific, do not forget to make sure the type you are casting to is the type of the object being referenced
* Retain the package structure from the previous assignment. EmployeeManager and EmployeeDriver can be compiled directly into the submission directory, that is, they do not belong to a package.

**Summary of Files**

You will need to six (6) files for this assignment.

* Employee.java
* HourlyEmployee.java
* SalaryEmployee.java
* CommissionEmployee.java
* EmployeeManager.java
* EmployeeDriver.java (provided)

**Required Elements:**

* Your program file must meet the programming standards defined for this course and contain the appropriate header defined for this course.
* Document your methods

# **Submitting Your Program Files:**

The assignment is automatically collected from your account on the “Loki” machine on the due date. You must put your source code in the correct directory in your account in order for this to work. Your directory name includes the course, section, semester and the assignment number. This is case-sensitive; **all alphas are upper-case**. This directory must be created in your home directory.

Submission Directory:

**CSCI-1620-1-F15-A2**