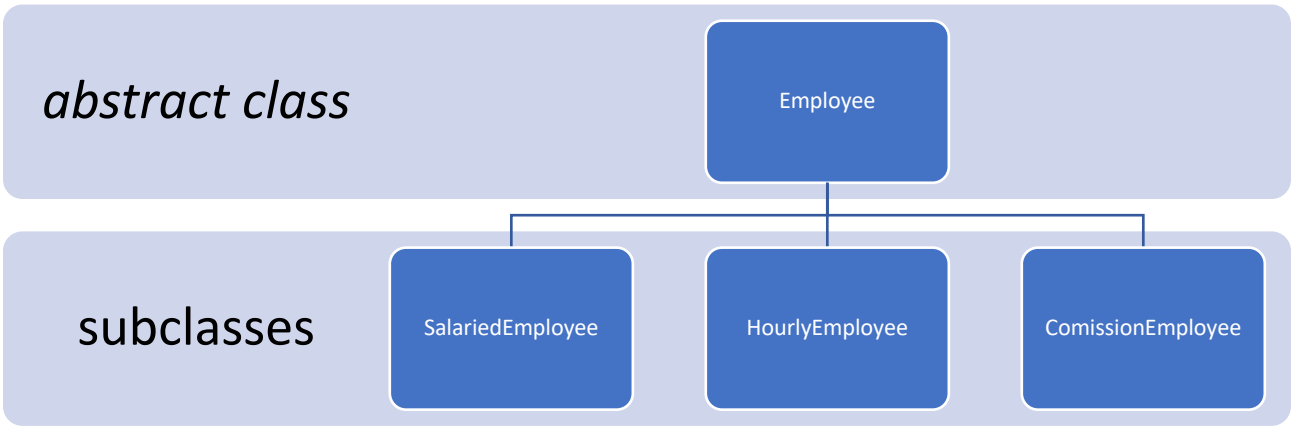
Overview:

Congratulations you have been promoted to Lead Software Engineer in your company! Your first task in your new position is to develop a new payroll system. As an experienced software engineer you immediately realize that there are three main types of employees that your system must account for: 1) Salaried, 2) Hourly, and 3) Commission-Based.

Based on the above you came up with the below class hierarchy. *Employee* must be an **abstract class** while *SalariedEmployee*, *HourlyEmployee*, and *ComissionEmployee* will be **subclasses** that extend *Employee*.



Part 01 (/20 marks)

Implement the UML diagram on page 4. Each class is graded out of 5 marks based on correctness and completeness.

Note: The implementation details for the *toString()* and *earnings()* methods can be found in Table 1 on page 2.

A series of helpful hints/clarification for each class follow:

Employee • *earnings* is an abstract method • *managedBy* - a reference to who the employee works for **(default is themselves)** • *numEmployee* is a static property, representing number of employees in the system • *count* is a static method, that returns the number of employees in the system • *equals(Object obj)* should only return true when both employees have the same SIN
SalariedEmployee • Inherits from *Employee* • *weeklySalary* is the salary per week received

ComissionEmployee

- Inherits from *Employee*
- *comissionRate* is rate of pay per sales dollar (i.e. 10% commission is 0.1)
- *grossSales* is the amount sold by employee
- **This class has a copy constructor**

HourlyEmployee

- Inherits from *Employee*
- *hourlyRate* is the pay per hour
- *hours* is number of hours per week worked

	earnings()	**toString()**
Employee	*abstract*	Name: *lastname, firstName* SIN: xxxxx
SalariedEmployee	Employee's weekly salary.	Type: Salaried Name: *lastname, firstName* SIN: xxxxx Weekly Salary: $*weeklySalary*
HourlyEmployee	Employee's hourly rate multiplied by number of hours for hours 0-40. Any time in excess of 40 hours is payed at time-and-a-half (hourlyRate * 1.5)	Type: Hourly Name: *lastname, firstName* SIN: xxxxx Hourly Rate: $*hourlyRate* Hours Worked: *hours*
ComissionEmployee	Employee's commission rate multiplied by gross sales made by the employee.	Type: Commission Name: *lastname, firstName* SIN: xxxxx Gross Sales $grossSales Commission Rate: *comissionRate*

Table 1: earnings() and toString() implementation details

Part 02 (/10 marks)

Create a main program called *payrollTester.* In the program create the following employees:

Type	Name	SIN	Managed By	Other Information
Salaried	Joe Francis	123456789		Weekly salary: $2500
Salaried	Samantha Hughes	444555666	Joe Francis	Weekly salary: $1400
Hourly	Kim Adams	888999000	Joe Francis	Hourly rate: $18.50 Hours worked: 42
Commission	Ryan Goodall	111222333	Samantha Hughes	Commission rate: 15% Gross sales: $9000

Add the newly created employees to an array or ArrayList<…> object. Next, iterate/loop through the employees in the structure and display output similar to the following:

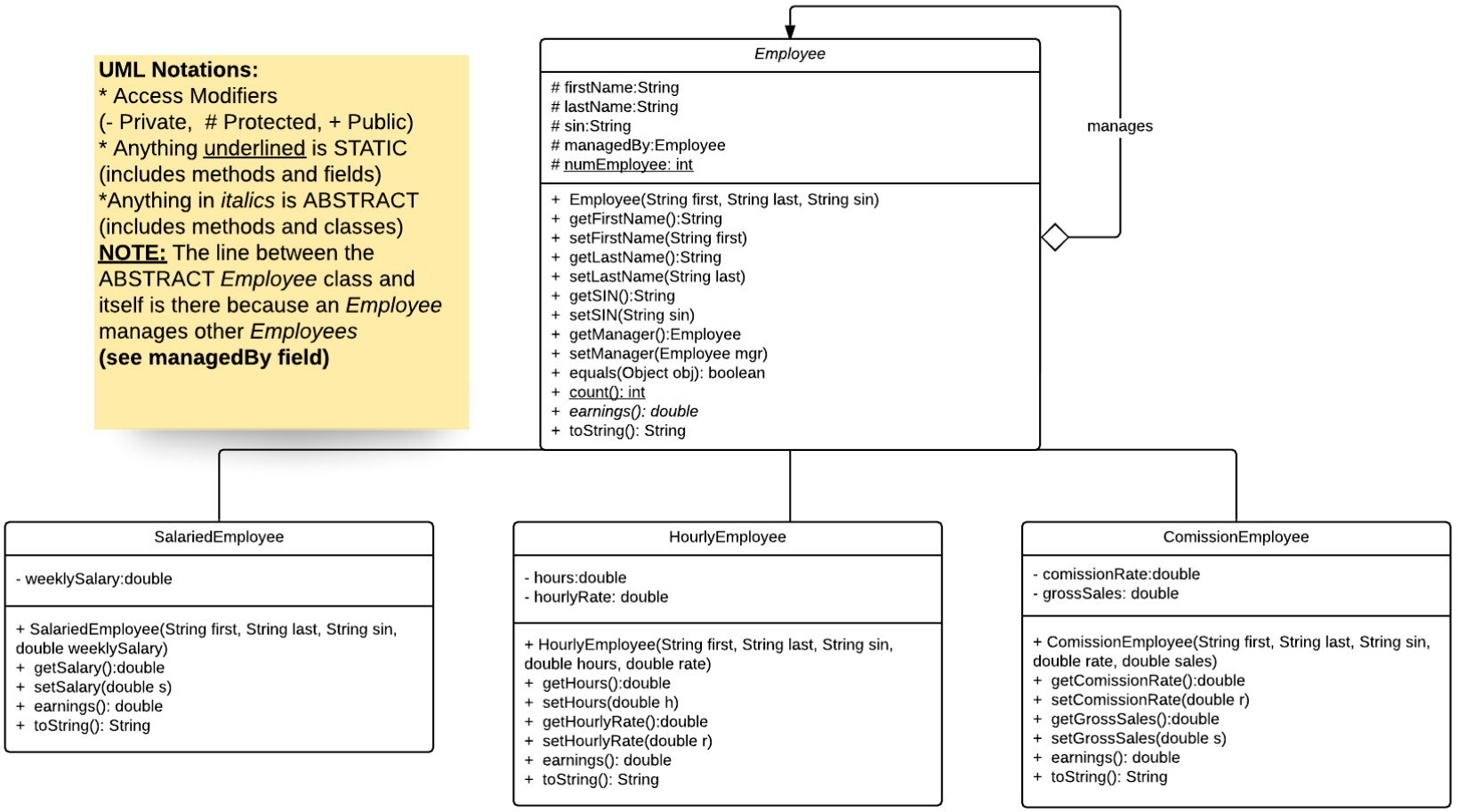
Type: Salaried
Name: Francis, Joe
SIN: 123456789
Weekly Salary: $2500
TOTAL: $2500

Type: Salaried
Name: Hughes, Samantha
SIN: 444555666
Weekly Salary: $1400
TOTAL: $1400

Type: Hourly
Name: Adams, Kim
SIN: 888999000
Hourly Rate: $*18.50*
Hours Worked: 42
TOTAL: $795.50

Type: Commission
Name: Goodall, Ryan
SIN: 111222333
Gross Sales $9000
Commission Rate: 0.10
TOTAL: $900

SUMMARY STATISTICS
Highest Paid Employee: Francis, Joe
Lowest Paid Employee: Kim, Adams
Number of salaried employees: 2
Number of Hourly employees: 1
Number of Commission employees: 1
Total for Pay Period: $5595.50


UML Notations:
* Access Modifiers
(- Private, # Protected, + Public)
* Anything underlined is STATIC
(includes methods and fields)
*Anything in italics is ABSTRACT
(includes methods and classes)
NOTE: The line between the
ABSTRACT Employee class and
itself is there because an Employee
manages other Employees
(see managedBy field)
Employee
firstName:String
lastName:String
sin:String
managedBy:Employee
numEmployee: int
+ Employee(String first, String last, String sin)
+ getFirstName():String
+ setFirstName(String first)
+ getLastName():String
+ setLastName(String last)
+ getSIN():String
+ setSIN(String sin)
+ getManager():Employee
+ setManager(Employee mgr)
+ equals(Object obj): boolean
+ count(): int
+ earnings(): double
+ toString(): String
manages
SalariedEmployee
- weeklySalary:double
+ SalariedEmployee(String first, String last, String sin, double weeklySalary)
+ getSalary():double
+ setSalary(double s)
+ earnings(): double
+ toString(): String
HourlyEmployee
- hours:double
- hourlyRate: double
+ HourlyEmployee(String first, String last, String sin, double hours, double rate)
+ getHours():double
+ setHours(double h)
+ getHourlyRate():double
+ setHourlyRate(double r)
+ earnings(): double
+ toString(): String
ComissionEmployee
- comissionRate:double
- grossSales: double
+ ComissionEmployee(String first, String last, String sin, double rate, double sales)
+ getComissionRate():double
+ setComissionRate(double r)
+ getGrossSales():double
+ setGrossSales(double s)
+ earnings(): double
+ toString(): String