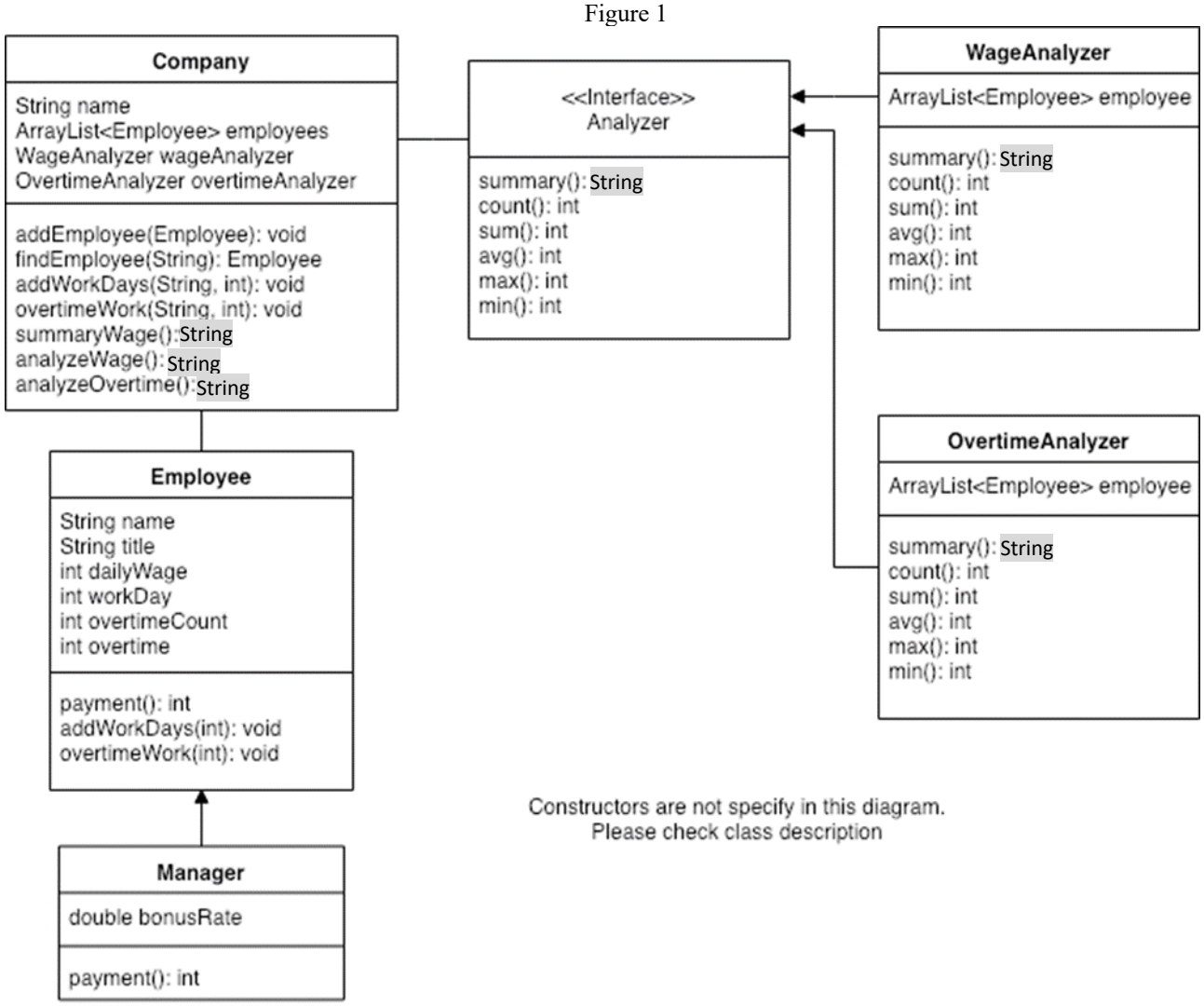
Assignment 1

Requirements:

- Create a Java project named assignGroup_HW1
- Read instructions and create classes needed. You are supposed to add 7 classes (1 interface + 5 required + 1 Tester) to the project.
- Your code must be properly formatted with sensible variable names! Refer to the text for code format examples.
- The instruction for Tester and output are for your reference.
- Make sure your classes correctly implement the public interfaces.

Following diagram shows the interface, inheritance, basic attributes and methods for each class.

Figure 1



Note that Employee is inherited by Manager, Employee DOESN'T inherit from Company, and Analyzer is implemented by **WageAnalyzer** and **OvertimeAnalyzer**. Adding getter or setter method when it is needed.

1. Create **Employee** class

Employee	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	name The name of this employee.
String	title The title of this employee.
int	dailyWage The basic daily wage of this employee.
int	workDay The actual work day(s).
int	overtimeCount The day(s) the employee works overtime.
int	overtime Total hours of working overtime.
Constructor	
Employee (String name, String title, int dailyWage) **Constructs an Employee object with given name, title, and dailyWage. And set remain attributes 0.**	
Instance methods	
int	payment() Calculate the wage and return the wage. (Wage = dailyWage * workDay + overtime * 150)
void	addWorkDays(int dayCount) Add work day(s) by given day count.
void	overtimeWork(int hour) Add overtime hour(s) by given hour(s), and increase overtimeCount by 1.

Test the employee class.

2. Create **Manager** class

Manager	
Modifier and type	**Method (or Variable) and description**
Instance variable	
double	bonusRate The manager can get extra wage (rate).
Constructor	
Manager (String name, String title, int dailyWage, double bonusRate) **Constructs an Manager object with given name, title, dailyWage and bonusRate. And set remain attributes 0.**	
Instance methods	
int	payment() Calculate the wage and return the wage. (Wage = (dailyWage * workDay + overtime * 150) * bonusRate) **Round it and cast the value to integer.**

Test the manager class.

3. Create **Company** class

Company	
Modifier and type	**Method (or Variable) and description**
Instance variable	
String	name

	The company's name.
ArrayList<Employee>	employees An ArrayList contains all employees in this company.
WageAnalyzer	wageAnalyzer A WageAnalyzer object.
OvertimeAnalyzer	overtimeAnalyzer An OvertimeAnalyzer object.
Constructor	
Company(String name) **Constructs a Company object with given name.**	
Instance methods	
void	addEmployee(Employee employee) Add an employee to the Employee ArrayList.
Employee	findEmployee(String name) Find the employee object in employees by the employee's name. If found, return the employee object. Otherwise, return null.
void	addWorkDays(String name, int day) Add an employee's work day(s) by given employee name and day count.
void	overtimeWork(String name, int hour) Add an employee's overtime hour(s) by given employee name and hour(s).
String	summaryWage() Return a formatted String about each employee's name, work day, overtime count, overtime hour(total), wage, and title.
String	analyzeWage() Invoke wageAnalyzer.summary() and return the output.
String	analyzeOvertime() Invoke overtimeAnalyzer.summary() and return the output.

Test the company object.

4. Create **Analyzer** interface

<<Interface>> **Analyzer**	
Modifier and type	**Method (or Variable) and description**
Abstract methods	
String	summary() Abstract method
int	count() Abstract method
int	sum() Abstract method
int	avg() Abstract method
int	max() Abstract method
int	min() Abstract method

5. Create **WageAnalyzer** class

WageAnalyzer	
Modifier and type	**Method (or Variable) and description**
Instance variable	
ArrayList<Employee>	employees An ArrayList contains all employees.
Constructor	
WageAnalyzer (ArrayList<Employee> employees) **Initialize the WageAnalyzer and set up the employees by given ArrayList.**	
Instance methods	
String	summary() Return a formatted String about the analysis of the wage of the company, including the number of employees, total wage, average wage, minimum wage, and maximum wage (See example output below).
int	count() Return the number of employees.
int	sum() Return the sum of all wage.
int	avg() Return the average of the wage.
int	max() Return the max value of the wage.
int	min() Return the min value of the wage.

Test the wage analyzer object.

6. Create **OvertimeAnalyzer** class

OvertimeAnalyzer	
Modifier and type	**Method (or Variable) and description**
Instance variable	
ArrayList<Employee>	employees An ArrayList contains all employees.
Constructor	
OvertimeAnalyzer (ArrayList<Employee> employees) **Initialize the OvertimeAnalyzer and set up the employees by given ArrayList.**	
Instance methods	
String	summary() Return a formatted String about the analysis of the working overtime of the company, including the number of working overtime employee, total hours, average hours, minimum hours, and maximum hours (See example output below).
int	count() Return the number of employees who working overtime.
int	sum() Return the sum of all overtime hours.
int	avg() Return the average of the overtime hours.
int	max() Return the max value of the overtime hours.
int	min()

	Return the min value of the overtime hours.

Test the overtime analyzer object.

Following Tester code and output are for your reference.

Tester

```
public class Tester {

	public static void main(String[] args) {
		Company company = new Company("NCCU");

		Employee emp1 = new Employee("Simon", "Staff", 1200);

		company.addEmployee(emp1);
		company.addEmployee(new Employee("Ding", "Staff",1100));
		company.addEmployee(new Manager("Wei", "Supervisor",1500, 1.1));

		company.addWorkDays("Simon", 5);
		company.addWorkDays("Simon", 5);
		company.overtimeWork("Simon", 1);
		company.overtimeWork("Simon", 1);
		company.overtimeWork("Simon", 1);

		company.addWorkDays("Ding", 7);;
		company.overtimeWork("Ding", 1);

		System.out.println("summarizeWage");
		System.out.println(company.summaryWage());
		System.out.println("analyzeWage");
		System.out.println(company.analyzeWage());
		System.out.println("analyzeOvertime");
		System.out.println(company.analyzeOvertime());
	}
}
```

Output	Format

summarizeWage
  Company: NCCU
  Name Work Day Overtime Count Overtime Hour(Total) Wage Title
  Simon 10 3 3 12450 Staff
  Ding 7 1 1 7850 Staff
  Wei 0 0 0 0 Supervisor
analyzeWage
  Total employees: 3
  Total wage: 20300
  Average wage: 6766
  Min wage: 0
  Max wage: 12450
 	
analyzeOvertime
  Total employees: 2
  Total hours: 4
  Average hours: 2
  Min hours: 1
  Max hours: 3
	10 10
	10 10 15 21 7 12
	10 10 15 21 7 12
	
	19 7
	
	19 7

Teammate evaluation: If you work in pair, please fulfill this form: https://forms.gle/AdjcZXjXhs2xP4Qm7

Test your assignment: Submit all "class" file, except for Tester via https://uoclab.nccu.edu.tw/oop

Submission: Submit your project as "zip (or rar) file" via WM5. No other submissions will be graded. Only one submission is needed.

Reminder: Please zip the whole project.

Deadline: 2020/3/15 23:59 (for both Mon56 and Tue23)