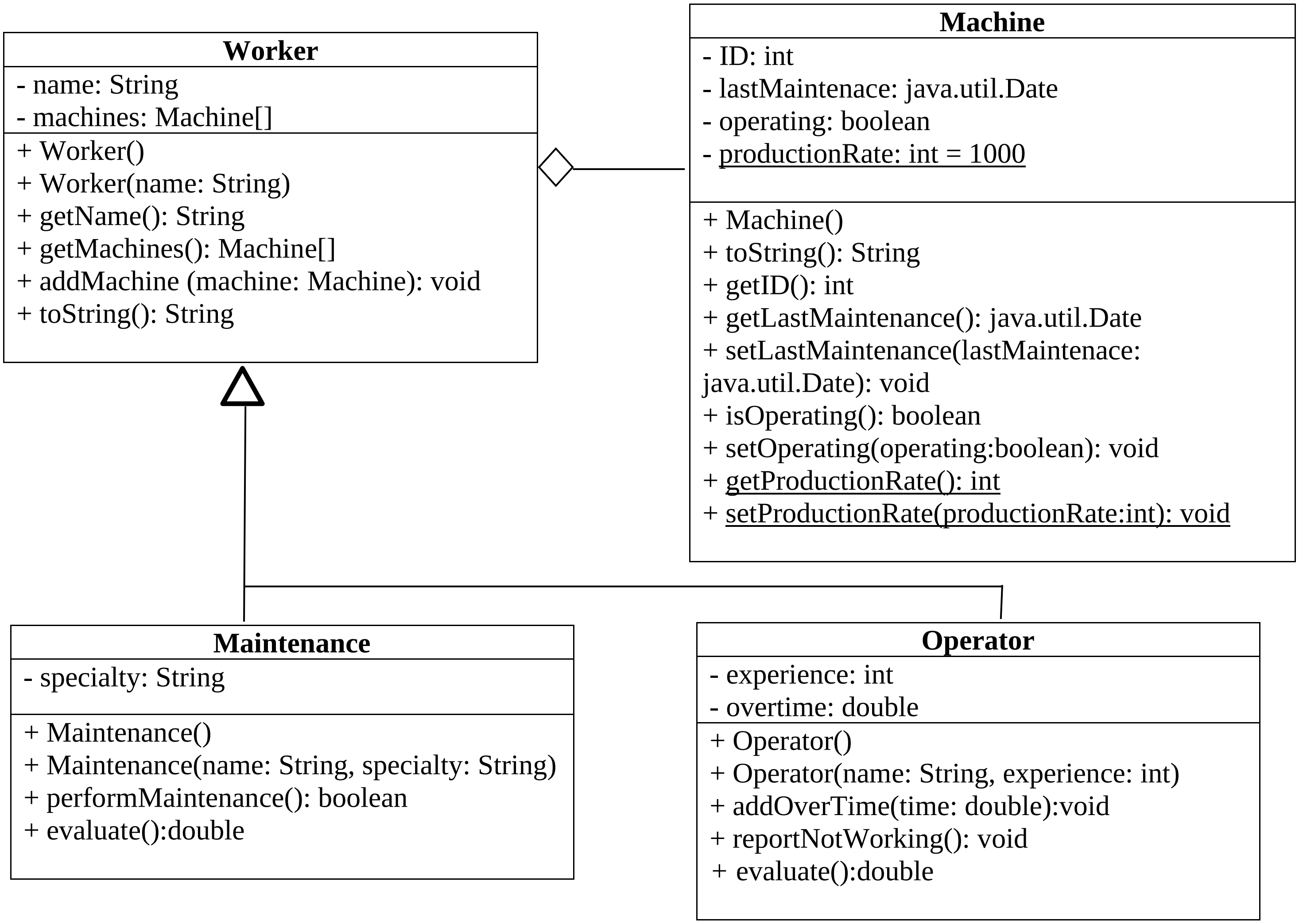
The University of Jordan
Faculty of Engineering and Technology
Department of Computer Engineering

Object-Oriented Problem Solving: CPE 342

- ❖ **Write a Java program that implements the classes shown in the following UML class diagram:**



Worker

- name: String
- machines: Machine[]

+ Worker()
+ Worker(name: String)
+ getName(): String
+ getMachines(): Machine[]
+ addMachine (machine: Machine): void
+ toString(): String

Machine

- ID: int
- lastMaintenace: java.util.Date
- operating: boolean
- productionRate: int = 1000

+ Machine()
+ toString(): String
+ getID(): int
+ getLastMaintenance(): java.util.Date
+ setLastMaintenance(lastMaintenace: java.util.Date): void
+ isOperating(): boolean
+ setOperating(operating:boolean): void
+ getProductionRate(): int
+ setProductionRate(productionRate:int): void

Maintenance

- specialty: String

+ Maintenance()
+ Maintenance(name: String, specialty: String)
+ performMaintenance(): boolean
+ evaluate():double

Operator

- experience: int
- overtime: double

+ Operator()
+ Operator(name: String, experience: int)
+ addOverTime(time: double):void
+ reportNotWorking(): void
+ evaluate():double

1. For the *Machine* (آلة) class:
   - The *no-arg constructor* must initialize the ID by reading it in a message dialog box.
   - The *lastMaintenance* date must be initialized to the current date.
   - The *operating* data field must be initialized to true.
   - The *toString* method must be overridden such that it returns a *String* that consists of info of the machine in the following format (each data field on a separate line). Dates info must display all date and time details.

   ID: __________
   Operating/Down
   Last maintenance: __________

   Operating if operating is true
   Down if operating is false

2. For the *Worker* (عامل) class:
   - The *no-arg constructor* must read the name from the user in an input dialog box.
   - The constructor *Worker(name: String),* must assign the *name* argument to the *name* data field.
   - Both constructors must initialize the *machines array* to be empty.
   - The *addMachine* method must add the passed *machine* object to the *machines array* such that the *array* is ordered ascendingly according to the machines IDs.
   - The *toString* method must be overridden such that it prints info of the worker as follows:

   Name: _______
   Number of machines responsible for: ____________

3. For the *Maintenance* (عامل صيانة) class:
   - The *no-arg constructor* must invoke the super class *no-arg constructor* to read the name, then read the *specialty* (تخصص) from the user in an input dialog box.
   - The constructor *Maintenance(name: String, specialty: String),* must pass the name to the appropriate super class constructor, and assign the *specialty* argument to the *specialty* data field.
   - The method *performMaintenance* must find the first machine of the *machines* for which this worker is responsible that is not operating (operating data field is false). This method must ask the user in a ConfirmDialogBox whether maintenance is performed successfully or not. If he answers yes, it must set the *LastMaintenanceDate* of the machine to the current date and the *operating* data field to true, then return true. If he answers no or cancel it performs nothing and returns false.

   Note: for this part you are required to learn more about confirm dialog boxes of the JOptionPane class, how they are invoked, how they are used, and what they return.

   - The *toString* method must be overridden such that it returns a *String* that consists of the String returned by the super class, combined with the specialty on a new line as follows:

   Specialty: ____________

   - The *evaluate* method must return a double value that represents the percentage of operating machines out of the total number of machines that this worker is responsible for.

4. **For the *Operator* (مشغل الالة) class:**

- The *no-arg constructor* must invoke the super class *no-arg constructor* to read the name, then read the *experience* (سنين الخبرة) from the user in an input dialog box to initialize the *experience*.

- The constructor *Operator(name: String, experience: double)*, must pass the name to the appropriate super class constructor, and initialize the *experience* with the passed argument.

- The *addOverTime* method must add the passed *time* argument to the *overTime* data field.

- The *reportNotWorking* method must ask the operator in a message dialog box about the ID of the machine which he wants to report that it is not working. After entering the ID, the method must search for the machine with the specified ID in the array *machines* and set its *operating* data field to false.

- The *toString* method must be overridden such that it returns a *String* that consists of the String returned by the super class, combined with the *experience* and *overTime* on new lines.

- The *evaluate* method must return a double value that represents the percentage of overtime to overall working time, assume that these are to be computed per week, working time without overtime is 40 hours.

5. **In your main method:**
   - **Create an array of 6 machines.**
   - **Print the *toString* method for each of the 6 machines in separate message dialog boxes.**
   - **Create two workers: one operator (whose name is Ahmad Saleem and his experience is 5 years) and one maintenance worker (whose name is Saad Mohammad and his specialty is *Mechanics)*.**
   - **Assign the six machines to both workers by adding them using the *addMachine* method.**
   - **Print the *toString* method for both workers in two separate message dialog boxes.**
   - **Test the remaining methods: *performMaintenance* of the Maintenance class, and *addOverTime, reportNotWorking* of the Operator class, and the evaluate method for both classes by invoking them using the two created worker objects and trying different scenarios. Print details of the machines and workers using the *toString* method to check the correctness of your methods implementation.**