Polymorphism - *having multiple forms*

**Objective:** The purpose of this exercise is to review Polymorphism with actual sample code.

**Example Driver: TestEmployees.java**

1. public class TestEmployees {
2.     public static void main(String[] args) {
3.         Employee[] employees = new Employee[4];
4.     }
5.     int employeeCount = 0;
6.     employees[employeeCount++] = new PartTime("Donald Duck", 20.00, 10.0);
7.     employees[employeeCount++] = new FullTime("Mickey Mouse", 1000.00);
8.     employees[employeeCount++] = new PartTime("Daisy Duck", 20.00, 5.5);
9.     employees[employeeCount++] = new FullTime("Minnie Mouse", 1000.00);
10.    System.out.println("Employee Name \t Employee Type \t Paycheck Amount");
11.    System.out.println("---------------------------------------------------------------------");
12.    for (int i = 0; i < employeeCount; i++) {
13.        System.out.print(employees[i]);
14.        System.out.format("$" + "%.2f", employees[i].pay());
15.    }
16.    }
17. }

---

**Compile all of our files**

> javac TestEmployees.java Employee.java FullTime.java PartTime.java

**Output**

> java TestEmployees

<table>
<thead>
<tr>
<th>Employee Name</th>
<th>Employee Type</th>
<th>Paycheck Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Donald Duck</td>
<td>Hourly</td>
<td>$200.00</td>
</tr>
<tr>
<td>Mickey Mouse</td>
<td>Salaried</td>
<td>$1000.00</td>
</tr>
<tr>
<td>Daisy Duck</td>
<td>Hourly</td>
<td>$110.00</td>
</tr>
<tr>
<td>Minnie Mouse</td>
<td>Salaried</td>
<td>$1000.00</td>
</tr>
</tbody>
</table>
**Base Class: Employee.java**

1. abstract public class Employee {

2. public Employee(String name) { setName(name); }

3. public String getName() { return new String(name); }

4. public void setName(String name) { this.name = new String(name); }

5. abstract public double pay();

6. public String toString() { return name; }

7. private String name;

8. }

**Derived Class #1: FullTime.java**

1. public class FullTime extends Employee {

2. public FullTime(String name, double salary) {
3.     super(name);
4.     setSalary(salary); }

5. public void setSalary(double salary) {
6.     this.salary = salary;
7. }

8. public double getSalary() { return salary; }

9. public double pay() { return salary; }

10. public String toString() {
11.     return super.toString() + "\t Salaried \t ";
}

12. double salary;

12. }
Derived Class #2: PartTime.java
1. public class PartTime extends Employee {

2.   public PartTime(String name, double rate, double hours) {
3.     super(name);
4.     setRate(rate);
5.     setHours(hours); }

6.   public void setRate(double rate) { this.rate = rate; }
7.   public void setHours(double hours) { this.hours = hours; }
8.   public double getRate() { return rate; }
9.   public double getHours() { return hours; }
10.  public double pay() { return rate * hours; }
11.  public String toString() { return super.toString() + " Hourly "; }

12.  private double rate;
13.  private double hours;
14.  }
Homework exercises (Part 1 – Written)
Download all of the files that we discussed today and modify them to explore the following questions.

1. Why doesn’t this line compile when I add it to the driver?:
   employees[employeeCount++] = new Employee("Minnie Mouse");

   Compiler error:
   > javac *.java
   TestEmployees.java:11: Employee is abstract; cannot be instantiated
   employees[employeeCount++] = new Employee("Minnie Mouse");

   Answer: The Employee class is _____________.

2. Is it okay to add these lines to the driver?:
   FullTime ftEmployee = new FullTime("Renee Bryce", 100.00);
   System.out.println(ftEmployee.getName());

   Answer: Yes! Here are the results of compiling and running the driver with this code:
   Compile:
   > javac *.java
   Run/Output:
   > java TestEmployees
   Renee Bryce

3. Can I change a name of an Employee?

   Answer: Yes! The “setName” method in our base class is ______________. Here is an example that you can place in the driver:
   
   Employee duck = new FullTime("Donald Duck", 100.00);
   System.out.println("Initial name: " + duck.getName());
   duck.setName("Super Duck");
   System.out.println("New name: " + duck.getName());

   Compile:
   > javac *.java
   Run/Output:
   > java TestEmployees
   Initial name: Donald Duck
   New name: Super Duck

4. If the “setName” method was private, would the code above work?
   Answer: No, using the code above, you would receive this error:
   TestEmployees.java:13: setName(java.lang.String) has private access in
   Employee duck.setName("Super Duck");

   You would need to make the setName method ______________.
5. Can I redesign this code so that the Employee class is not abstract, but I can override the “pay” method?

   Answer: Yes! Here is an example:

Modified Employee.java file:

// This class is no longer abstract, nor is the pay method!

1. public class Employee {
2.     public Employee(String name) { setName(name); }
3.     public String getName() { return new String(name); }
4.     private void setName(String name) { this.name = new String(name); }
5.     public double pay()
6.         System.out.println("Employee not set up for payroll.");
7.         System.out.println("Please define as a fulltime or part-time employee.");
8.         return 0.0;
9.     public String toString() { return name; }
10.    private String name;
11. }

Use this code in the driver:

   Employee baseExampleEmployee = new Employee("Renee");
   System.out.println("Employee name: " + baseExampleEmployee.toString());
   System.out.println("Salary: " + baseExampleEmployee.pay());

Compile:
   > javac *.java
Run/Output:
   > java TestEmployees
   Employee name: Renee
   Employee not set up for payroll.
   Please define as a fulltime or part-time employee.
   Salary: 0.0
6. Can you change an Employee to a FullTime employee? If so, provide the code for this. If not, what is the compiler error?

*Answer:*

7. Place this code in your driver. Does the code compile? If so, what is the output?

```java
FullTime employee1 = new FullTime("Mickey Mouse", 1.0);
FullTime employee2 = new FullTime("Minnie Mouse", 2.0);

employee1 = employee2;
System.out.println("Employee1: " + employee1);
System.out.println("Employee2: " + employee2);

*Answer:*
```

8. Place this code in your driver. Does code compile? If so, what is the output?

```java
FullTime employee1 = new FullTime("Mickey Mouse", 1.0);
FullTime employee2 = new FullTime("Minnie Mouse", 2.0);

employee1 = employee2;
System.out.println("Employee1: " + employee1);
System.out.println("Employee2: " + employee2);

PartTime employee3 = new PartTime("Donald Duck", 10.0, 2.0);
System.out.println("Employee3: " + employee3);
employee3 = employee1;
System.out.println("Employee3: " + employee3);

*Answer:*
```
Homework exercise (Part 2 – Coding)

Download the code from this exercise and create a new class, SalesEmployee. This class will extend the Employee class, similar to our examples of the FullTime and PartTime classes. The SalesEmployee class should contain:

1. A constructor that takes an Employee’s name, number of widgets sold, and commission rate per widget.

2. The pay method will compute their weekly pay as:
   \[(\text{number of widgets sold}) \times (\text{commission rate per widget})\]

3. Include appropriate instance variables and corresponding set/get methods

4. The toString method should print the Employee name, followed by the word “Sales” since they are a sales employee.

5. Provide examples in the driver to show that your code works.