Module Six - Assignment Submission

The Assignment

> This assignment will use the Employee class that you developed for assignment 6. Design two sub-classes of Employee...SalariedEmployee and HourlyEmployee. A salaried employee has an annual salary attribute. An hourly employee has an hourly pay rate attribute, an hours worked attribute, and an earnings attribute. An hourly employee that works more than 40 hours in a week gets paid at 1.5 times their hourly pay rate for the hours over 40 hours. You will decide how to implement constructors, getters, setters, and any other methods that might be necessary.
>
> 1. Draw a UML diagram for the classes.
> 2. Implement the classes, and write a test program that creates a salaried employee and two hourly employees. One of the hourly employees should have hours worked set to less than 40 and one should have hours worked set to more than 40. The test program should display all attributes for the three employees. To keep things simple, the employee classes don't need to do any editing.

Design

Straightforward implementation again. I cleaned up some of the style issues I had in module 6.

No real noted on design here, other than my choice to use a double for salary so we can do penny denominations, which does assume an American/Euro employee with 2 decimals max of hourly/salary monetary value.

Implementation

```
package module7;

// Class which implements the test of the employees

public class testEmployee {
    public static void main(String[] args) {
        // Create a salaried employee
        salariedEmployee salariedEmployee = new salariedEmployee(
            new name("John", "Doe"),
            new address("123 Main St", "Baltimore", "CA", 21218),
            new date(1, 1, 1975),
            1,
            50000
        );

        // Create two hourly employees
        hourlyEmployee hourlyEmployee1 = new hourlyEmployee(
            new name("Jane", "Smith"),
            new address("456 Oak St", "Boston", "MA", 67890),
            new date(2, 2, 1975),
```

```
            2,
            20,
            35
        );

        hourlyEmployee hourlyEmployee2 = new hourlyEmployee(
            new name("Bob", "Johnson"),
            new address("789 Pine St", "Watertown", "FL", 11223),
            new date(3, 3, 1975),
            3,
            20,
            45
        );

        // Display attributes for the salaried employee
        System.out.println("Salaried Employee:");
        System.out.println(salariedEmployee);
        System.out.println();

        // Display attributes for the first hourly employee
        System.out.println("Hourly Employee 1:");
        System.out.println(hourlyEmployee1);
        System.out.println();

        // Display attributes for the second hourly employee
        System.out.println("Hourly Employee 2:");
        System.out.println(hourlyEmployee2);
    }
} // end class TestEmployee

package module7;

/**
 * The Employee class represents an employee with basic information
 * such as name, address, birthday, and employee number. This class
 * provides getter and setter methods for the attributes and includes
 * an overridden toString method for displaying the employee's details.
 */

public class employee {
    // Constructor
    public employee(name name, address address, date birthday, int
employeeNumber) {
        this.name = name;
        this.address = address;
        this.birthday = birthday;
        this.employeeNumber = employeeNumber;
    }

    // Getters
    public name getName() {
        return name;
    }
```

```
    public address getAddress() {
        return address;
    }

    public date getBirthday() {
        return birthday;
    }

    public int getEmployeeNumber() {
        return employeeNumber;
    }

    // Setters
    public void setName(name name) {
        this.name = name;
    }

    public void setAddress(address address) {
        this.address = address;
    }

    public void setBirthday(date birthday) {
        this.birthday = birthday;
    }

    public void setEmployeeNumber(int employeeNumber) {
        this.employeeNumber = employeeNumber;
    }

    // toString method
    public String toString() {
        return "Name: " + name.toString() + "\n" +
               "Address: " + address.toString() + "\n" +
               "Birthday: " + birthday.getDay() + "/" + birthday.getMonth()
+ "/" + birthday.getYear() + "\n" +
               "Employee Number: " + employeeNumber;
    }

    // equals method
    public boolean equals(employee other) {
        if(other == null) {
            return false;
        }
        return this.name.equals(other.name) &&
               this.address.equals(other.address) &&
               this.birthday.equals(other.birthday) &&
               this.employeeNumber == other.employeeNumber;
    }

    // Class Members
    protected name name;
    protected address address;
    protected date birthday;
    protected int employeeNumber;
```

```
} // end class employee

package module7;

// Class HourlyEmployee

/**

* The HourlyEmployee class extends the Employee class and represents

 * an employee who is paid hourly. This class contains additional
attributes

 * for hourly pay rate, hours worked, and earnings. It includes a method

 * to calculate earnings, including overtime pay for hours worked beyond
40.

 */

public class hourlyEmployee extends employee {


    /**
     * Constructor to initialize an HourlyEmployee object with the given
attributes.
     *
     * @param name The name of the employee.
     * @param address The address of the employee.
     * @param birthday The birthdate of the employee.
     * @param employeeNumber The employee's unique identifier.
     * @param hourlyPayRate The hourly pay rate for the employee.
     */
    public hourlyEmployee(name name, address address, date birthday, int
employeeNumber, double hourlyWage, int hoursWorked) {
        super(name, address, birthday, employeeNumber);
        this.hourlyWage = hourlyWage;
        this.hoursWorked = hoursWorked;
        this.earnings =  this.getEarnings();
    }

    // Getters
    public double getHourlyWage() {
        return hourlyWage;
    }

    public int getHoursWorked() {
        return hoursWorked;
    }

    // Method to calulate earnings per the hourly wage and hours worked
    // If hours worked is greater than 40, calculate overtime pay
    public double getEarnings() {
        if (hoursWorked <= 40) {
```

```
            earnings = hourlyWage * hoursWorked;
        } else {
            earnings = (hourlyWage * 40) + ((hoursWorked - 40) * hourlyWage
* 1.5);
        }
        return earnings;
    }

    // Setters
    public void setHourlyWage(double hourlyWage) {
        this.hourlyWage = hourlyWage;
    }

    public void setHoursWorked(int hoursWorked) {
        this.hoursWorked = hoursWorked;
    }

    // toString method
    public String toString() {
        return super.toString() + "\n" +
               "Hourly Wage: " + hourlyWage + "\n" +
               "Hours Worked: " + hoursWorked;
    }

    // Class Members
    private double hourlyWage;
    private int hoursWorked;
    private double earnings;

} // end class hourlyEmployee

package module7;

// Class SalariedEmployee
/**
 * The SalariedEmployee class extends the Employee class and represents
 * an employee with a fixed annual salary. This class provides a method
 * to calculate the salary and displays the details of the salaried
employee.
 */

public class salariedEmployee extends employee {
    // Annual fixed salary

    /**
     * Constructor to initialize a SalariedEmployee object with the given
attributes.
     *
     * @param name The name of the employee.
     * @param address The address of the employee.
     * @param birthday The birthdate of the employee.
     * @param employeeNumber The employee's unique identifier.
     * @param annualSalary The annual salary of the employee.
     */
```

```
    public salariedEmployee(name name, address address, date birthday, int
employeeNumber, double salary) {
        super(name, address, birthday, employeeNumber);
        this.salary = salary;
    }

    // Getters
    public double getSalary() {
        return salary;
    }

    // Setters
    public void setSalary(double salary) {
        this.salary = salary;
    }

    // toString method
    public String toString() {
        return super.toString() + "\n" +
               "Salary: " + salary;
    }

    // Class Members
    private double salary;

} // end class salariedEmployee

/**
 * Class which implements a basic name
 *
 * Consists of a first name and a last name
 */
package module7;

public class name {


    // Constructor
    // Throws an error if impossible values are passed
    // Doesn't guarantee valid name
    public name(String firstName, String lastName) {
        if(firstName == null || firstName.isEmpty()) {
            throw new IllegalArgumentException("First name empty");
        }
        if(lastName == null || lastName.isEmpty()) {
            throw new IllegalArgumentException("Last name empty");
        }
        this.firstName = firstName;
        this.lastName = lastName;
    }

    // Getters
    public String getFirstName() {
```

```
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    // Setters
    public void setFirstName(String firstName) {
        if(firstName == null || firstName.isEmpty()) {
            throw new IllegalArgumentException("First name empty");
        }
        this.firstName = firstName;
    }

    public void setLastName(String lastName) {
        if(lastName == null || lastName.isEmpty()) {
            throw new IllegalArgumentException("Last name empty");
        }
        this.lastName = lastName;
    }

    // toString method
    public String toString() {
        return firstName + " " + lastName;
    }

    // equals method
    public boolean equals(name other) {
        if(other == null) {
            return false;
        }
        return this.firstName.equals(other.firstName) &&
this.lastName.equals(other.lastName);
    }

    // Class Members
    private String firstName;
    private String lastName;

} // end class name

/**
 * Class which represents a date
 *
 * Consists of a month, day, and year
 * and provides methods to manipulate and display the date
 */
package module7;

public class date {

    // Constructor
    // Throws an error if impossible values are passed
```

```
    // Doesn't guarantee valid date
    public date(int date, int month, int year) {
        if(date < 1 || date > 31) {
            throw new IllegalArgumentException("Day must be between 1 and
31");
        }
        if(month < 1 || month > 12) {
            throw new IllegalArgumentException("Month must be between 1 and
12");
        }
        if(year < 1900) {
            throw new IllegalArgumentException("Birthday must be after
1900");
        }
        if(year > 2020) {
            throw new IllegalArgumentException("Year must be before 2020");
        }

        this.day = date;
        this.month = month;
        this.year = year;
    }


    // Getters
    public int getDay() {
        return day;
    }

    public int getMonth() {
        return month;
    }

    public int getYear() {
        return year;
    }

    // Setters
    public void setDay(int day) {
        if(day < 1 || day > 31) {
            throw new IllegalArgumentException("Day must be between 1 and
31");
        }
        this.day = day;
    }

    public void setMonth(int month) {
        if(month < 1 || month > 12) {
            throw new IllegalArgumentException("Month must be between 1 and
12");
        }
        this.month = month;
    }
```

```
    public void setYear(int year) {
        if(year < 1900) {
            throw new IllegalArgumentException("Birthday must be after
1900");
        }
        if(year > 2020) {
            throw new IllegalArgumentException("Year must be before 2020");
        }
        this.year = year;
    }

    // toString method
    @Override
    public String toString() {
        return String.format("%02d/%02d/%04d", day, month, year);
    }

    // equals method
    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        date other = (date) obj;
        return day == other.day && month == other.month && year ==
other.year;
    }

    // Class Members
    private int day;
    private int month;
    private int year;
} // end class date

/**
 * Class which implements an american address
 *
 * Consists of a street address, city, state, and zip code
 * and provides methods to manipulate and display the address
*/
package module7;

public class address {

    // Constructor
    // Throws an error if impossible values are passed
    // Doesn't guarantee valid address
    public address(String streetAddress, String city, String state, int
zipCode) {
        this.streetAddress = streetAddress;
        this.city = city;
        this.state = state;
        this.zipCode = zipCode;
    }
```

```
    // Getters
    public String getStreetAddress() {
        return streetAddress;
    }

    public String getCity() {
        return city;
    }

    public String getState() {
        return state;
    }

    public int getZipCode() {
        return zipCode;
    }

    // Setters
    public void setStreetAddress(String streetAddress) {
        this.streetAddress = streetAddress;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public void setState(String state) {
        this.state = state;
    }

    public void setZipCode(int zipCode) {
        this.zipCode = zipCode;
    }
    // toString method
    public String toString() {
        return streetAddress + "\n" + city + ", " + state + " " + zipCode;
    }

    // Class Members
    private String streetAddress;
    private String city;
    private String state;
    private int zipCode;
} // end class address
```

Output

```
cd /home/arkosh/repos/601_201 ; /usr/bin/env /usr/lib/jvm/java-21-openjdk-
amd64/bin/java -XX:+ShowCodeDetailsInExceptionMessages -cp
```

```
/home/arkosh/.config/Code/User/workspaceStorage/031fd3b3b4e8843882e51998cfa
457cb/redhat.java/jdt_ws/601_201_8d1ee059/bin module7.TestEmployee
Salaried Employee:
Name: John Doe
Address: 123 Main St
Baltimore, CA 21218
Birthday: 1/1/1975
Employee Number: 1
Salary: 50000.0

Hourly Employee 1:
Name: Jane Smith
Address: 456 Oak St
Boston, MA 67890
Birthday: 2/2/1975
Employee Number: 2
Hourly Wage: 20.0
Hours Worked: 35

Hourly Employee 2:
Name: Bob Johnson
Address: 789 Pine St
Watertown, FL 11223
Birthday: 3/3/1975
Employee Number: 3
Hourly Wage: 20.0
Hours Worked: 45
```