Module Six - Assignment Submission

The Assignment

> Write a program that models an employee. An employee has an employee number, a name, an address, and a hire date. A name consists of a first name and a last name. An address consists of a street, a city, a state (2 characters), and a 5-digit zip code. A date consists of an integer month, day and year. All fields are required to be non-blank. The Date fields should be reasonably valid values (ex. month 1-12, day 1-31, year > 1900 and < 2020). Issue appropriate error messages when incorrect data is entered
>
> Create an Employee, a Name, an Address, and a Date classes in your solution. You may use the Date class from the lectures or define your own. The Java-supplied Date class cannot be used. Provide appropriate class constructors, getter methods, setter methods, toString() and any other methods you think are necessary. To keep things simple, your classes don't need to do any editing of data. The classes should not do any direct user input/output. All user input/output will be done in the main method
>
> Your program should prompt the user to enter data for several employees, store the data in an array of type Employee, and then display the data

Design

Implementing this program is very straightforward. For the employee class, the assignment wrotely spells out the requirements for the subclasses. The implementations just use a protected version of the class members described by the program, and then the appropriate error throws around it, and then finally a toString method and an equals method to do some basic functions outside of the get and set methods.

The caller program is just a for loop of prompts to the end user, which collected the necessary information for a few employees, and then prints out each user.

Not much else in the manner of design work on this one.

Implementation

```
package module6;

import java.util.Scanner;

public class assignmentSix {
    // Program entry point
    public static void main(String[] args) {
        // Declare and initialize variables
        // Prompt user to start the program
        System.out.println("Welcome to employee manager 3000!");
        System.out.println("Please enter the number of employees you would
like to enter: ");
        Scanner input = new Scanner(System.in);
```

```
        int numEmployees = input.nextInt();
        input.nextLine(); // Consume newline
        employee[] employees = new employee[numEmployees];

        // Create an instance of assignmentSix
        assignmentSix instance = new assignmentSix();

        // Loop through and get employee data
        for (int i = 0; i < numEmployees; i++) {
            System.out.println("Enter employee " + (i + 1) + " data: ");
            employees[i] = instance.getEmployee(input);
        }

        // Print out employee data
        System.out.println("Employee data entered: ");
        for (int i = 0; i < numEmployees; i++) {
            System.out.println("\nEmployee " + (i + 1) + ": ");
            System.out.println(employees[i].toString());
        }
        // Close the scanner
        input.close();
        System.out.println("Thank you for using employee manager 3000!");
        System.out.println("Goodbye!");
    } // end main

    // Method which gets employee data from standard in
    private employee getEmployee(Scanner input) {

        employee toReturn;
        name name;
        address address;
        date birthday;
        int employeeNumber;

        // Get Employee Name
        System.out.print("Enter employee name.\n");
        System.out.print("First Name: ");
        String firstName = input.nextLine();
        System.out.print("Last Name: ");
        String lastName = input.nextLine();
        name = new name(firstName, lastName);

        // Get Employee ID
        System.out.print("Enter employee ID: ");
        employeeNumber = input.nextInt();
        input.nextLine(); // Consume newline

        // Get Address
        System.out.print("Enter employee address.\n Street: ");
        String street = input.nextLine();
        System.out.print(" City: ");
        String city = input.nextLine();
        System.out.print(" State: ");
        String state = input.nextLine();
```

```
        System.out.print(" Zip: ");
        int zip = input.nextInt();

        address = new address(street, city, state, zip);

        // Get Birthday
        System.out.print("Enter employee birthday. MM/DD/YYYY\n Month: ");
        int month = input.nextInt();
        System.out.print(" Day: ");
        int day = input.nextInt();
        System.out.print(" Year: ");
        int year = input.nextInt();
        birthday = new date(day, month, year);
        input.nextLine(); // Consume newline

        // Make Employee and Return
        toReturn = new employee(name, address, birthday, employeeNumber);

        return toReturn;
    }   // end getEmployee
} // end class assignmentSix
```

```
package module6;

public class employee {
    protected
        name name;
        address address;
        date birthday;
        int employeeNumber;

    // 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;
    }
} // end class employee
```

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

```java
package module6;

public class date {
    private int day;
    private int month;
    private int year;

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

    // equals method
    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;
    }
} // 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 module6;

public class address {

    private String streetAddress;
    private String city;
    private String state;
    private int zipCode;
```

```
    // 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;
    }

} // end class address
```

```java
/**
 * Class which implements a basic name
 *
 * Consists of a first name and a last name
 */
package module6;

public class name {
    // vars
    private String firstName;
    private String lastName;

    // 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);
    }
} // end class name
```

Output

```
Script started on 2025-03-02 21:43:06-05:00 [TERM="xterm-256color"
TTY="/dev/pts/2" COLUMNS="248" LINES="20"]
□[?2004h□]0;arkosh@greyskull:
~/repos/601_201□□[01;32markosh@greyskull□[00m:□[01;34m~/repos/601_201□[00m$
zip /home/arkosh/pics1.zip *□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□sudo journalctl -xe
| grep -i "sda"□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□[7Pzip
/home/arkosh/pics1.zip *□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□[K□^C□[?2004l
□[?2004h□[?2004l

□[?2004h□]0;arkosh@greyskull:
~/repos/601_201□□[01;32markosh@greyskull□[00m:□[01;34m~/repos/601_201□[00m$
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 module6.assignmentSix
□[?2004l
Welcome to employee manager 3000!
Please enter the number of employees you would like to enter:
2
Enter employee 1 data:
Enter employee name.
First Name: Jorkou□ □□ □us □ □
Last Name: Jenkins
Enter employee ID: 10101010
Enter employee address.
 Street: 1000 □ □□ □ Grand Street
 City: T□ □New York □ □
 State: NY □ □
 Zip: 12345
Enter employee birthday. MM/DD/YYYY
 Month: 09
 Day: 10
 Year: 2011
Enter employee 2 data:
Enter employee name.
First Name: Jackie □ □
Last Name: Daytona
```

```
Enter employee ID: 335821
Enter employee address.
 Street: Big B□ □□ □□ □□ □□ □12 Bar STretch□ □□ □□ □□ □□ □□ □r□ □treet
 City: Tuscon □ □
 State: AZ
 Zip: 0213□ □□ □□ □□ □54321
Enter employee birthday. MM/DD/YYYY
 Month: 05
 Day: 10
 Year: 1800
Exception in thread "main" java.lang.IllegalArgumentException: Birthday
must be after 1900
    at module6.date.<init>(date.java:25)
    at module6.assignmentSix.getEmployee(assignmentSix.java:79)
    at module6.assignmentSix.main(assignmentSix.java:23)
□[?2004h□]0;arkosh@greyskull:
~/repos/601_201□□[01;32markosh@greyskull□[00m:□[01;34m~/repos/601_201□[00m$
zip /home/arkosh/pics1.zip *□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□[K^C□[?2004l
□[?2004h□[?2004l

□[?2004h□]0;arkosh@greyskull:
~/repos/601_201□□[01;32markosh@greyskull□[00m:□[01;34m~/repos/601_201□[00m$
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 module6.assignmentSix
□[?2004l
Welcome to employee manager 3000!
Please enter the number of employees you would like to enter:
1
Enter employee 1 data:
Enter employee name.
First Name: Javkie□ □□ □□ □□ □ckie
Last Name: Daytona
Enter employee ID: 1234567
Enter employee address.
 Street: 12 Bar STree□ □□ □□ □□ □treet
 City: Tuscon □ □
 State: AZ
 Zip: 54321
Enter employee birthday. MM/DD/YYYY
 Month: 10
 Day: 31
 Year: 1900
Employee data entered:

Employee 1:
Name: Jackie Daytona
Address: 12 Bar Street
Tuscon, AZ 54321
Birthday: 31/10/1900
Employee Number: 1234567
Thank you for using employee manager 3000!
Goodbye!
```

```
□[?2004h□]0;arkosh@greyskull:
~/repos/601_201□□[01;32markosh@greyskull□[00m:□[01;34m~/repos/601_201□[00m$
exit
□[?2004l
exit

Script done on 2025-03-02 21:45:44-05:00 [COMMAND_EXIT_CODE="0"]
```