Selected files

4 printable files

AppTest.java
Employee.java
Project.java
ProjectPrinter.java

AppTest.java

```java
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Locale;

public class AppTest {
    public static void main(String[] args) throws ParseException {

        // initialize date formatter
        SimpleDateFormat sdf = new SimpleDateFormat("MM/dd/yyyy", Locale.ENGLISH);

        // create project object
        Project project = new Project();
        // initialize project object with some values
        project.setProjectId("1");
        project.setStartDate(sdf.parse("06/24/2017"));
        project.setEndDate(sdf.parse("06/30/2017"));

        ArrayList<Employee> employeeArrayList = new ArrayList<>();
        employeeArrayList.add(new Employee("1", "ABC", 200, 2, 3));
        employeeArrayList.add(new Employee("2", "XYZ", 300, 3, 3));
        employeeArrayList.add(new Employee("3", "BCD", 350, 2, 1));
        employeeArrayList.add(new Employee("4", "PQR", 350, 1, 1));

        project.setListOfEmployees(employeeArrayList);

        System.out.printf("\nProject toString method result : \n%s\n", project);
        System.out.println("Project Budget : " + project.estimateBudget());
        System.out.println();
        System.out.println("Project Details From Project Printer : ");

        ProjectPrinter projectPrinter = new ProjectPrinter(project);
        Thread t1 = new Thread(projectPrinter);
        t1.start();
    }
}
```

Employee.java

```java
//employee class implements Comparable interface
public class Employee implements Comparable<Employee> {
    // instance variables
    private String employeeId;
    private String employeeName;
    private Integer salaryPerHour;
    private Integer noOfLeavingDay;
    private Integer noOfTravelDay;

    // parameterized constructor
```

```
    public Employee(String employeeId, String employeeName, int salaryPerHour, int noOfLeavingDay, int noOfTravelDay) {
        this.employeeId = employeeId;
        this.employeeName = employeeName;
        this.salaryPerHour = salaryPerHour;
        this.noOfLeavingDay = noOfLeavingDay;
        this.noOfTravelDay = noOfTravelDay;
    }

    // getter method
    public String getEmployeeId() {
        return employeeId;
    }

    // setter method
    public void setEmployeeId(String employeeId) {
        this.employeeId = employeeId;
    }

    // getter method
    public String getEmployeeName() {
        return employeeName;
    }

    // setter method
    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }

    // getter method
    public int getSalaryPerHour() {
        return salaryPerHour;
    }

    // setter method
    public void setSalaryPerHour(int salaryPerHour) {
        this.salaryPerHour = salaryPerHour;
    }

    // getter method
    public int getNoOfLeavingDay() {
        return noOfLeavingDay;
    }

    // setter method
    public void setNoOfLeavingDay(int noOfLeavingDay) {
        this.noOfLeavingDay = noOfLeavingDay;
    }

    // getter method
    public int getNoOfTravelDay() {
        return noOfTravelDay;
    }

    // setter method
    public void setNoOfTravelDay(int noOfTravelDay) {
        this.noOfTravelDay = noOfTravelDay;
    }

    // Calculate weekly salary
```

```
70      public Integer calculateWeeklySalary() {
71          return salaryPerHour * 8 * (5 - noOfLeavingDay + noOfTravelDay / 2);
72      }
73
74      // Implemented method for comparable interface
75      @Override
76      public int compareTo(Employee o) {
77
78          // Compare based on noOfTravelDay
79          int rankOfFirstObj = this.noOfTravelDay - o.getNoOfTravelDay();
80
81          // If noOfTravelDay is the same for both objects
82          if (rankOfFirstObj == 0) {
83              // Compare based on noOfLeavingDay
84              rankOfFirstObj = this.noOfLeavingDay - o.getNoOfLeavingDay();
85          }
86
87          // Return the rank
88          return rankOfFirstObj;
89      }
90
91      @Override
92      public String toString() {
93          return "[Name:" + employeeName + " - " + "Salary Per Hour:" + salaryPerHour + " - "
    + "Weekly Salary: "
94                  + calculateWeeklySalary() + "]";
95      }
96  }
97
```

Project.java

```
1   import java.text.ParseException;
2   import java.text.SimpleDateFormat;
3   import java.util.ArrayList;
4   import java.util.Collections;
5   import java.util.Date;
6   import java.util.concurrent.TimeUnit;
7
8   public class Project {
9       // instance variables
10      private String projectId;
11      private Date startDate;
12      private Date endDate;
13      private ArrayList<Employee> listOfEmployees;
14
15      // default constructor
16      public Project() {
17
18      }
19
20      // parameterized constructor
21      public Project(String projectId, Date startDate, Date endDate, ArrayList<Employee>
    listOfEmployees) {
22          this.projectId = projectId;
23          this.startDate = startDate;
24          this.endDate = endDate;
25          this.listOfEmployees = listOfEmployees;
26      }
27
```

```java
    // getter method
    public String getProjectId() {
        return projectId;
    }

    // setter method
    public void setProjectId(String projectId) {
        this.projectId = projectId;
    }

    // getter method
    public Date getStartDate() {
        return startDate;
    }

    // setter method
    public void setStartDate(Date startDate) {
        this.startDate = startDate;
    }

    // getter method
    public Date getEndDate() {
        return endDate;
    }

    // setter method
    public void setEndDate(Date endDate) {
        this.endDate = endDate;
    }

    // getter method
    public ArrayList<Employee> getListOfEmployees() {
        return listOfEmployees;
    }

    // setter method
    public void setListOfEmployees(ArrayList<Employee> listOfEmployees) {
        this.listOfEmployees = listOfEmployees;
    }

    // estimates budget based on employee's SalaryPerHour
    public Integer estimateBudget() {
        // find no of days between start and end date
        long diffInMillies = Math.abs(endDate.getTime() - startDate.getTime());
        long NO_OF_DAYS = TimeUnit.DAYS.convert(diffInMillies, TimeUnit.MILLISECONDS);

        // assuming
        long TOTAL_WORKING_HOURS = 8;
        long total = 0;

        // loop through listOfEmployees
        for (int i = 0; i < listOfEmployees.size(); i++) {
            Employee employee = listOfEmployees.get(i);
            total += NO_OF_DAYS * TOTAL_WORKING_HOURS * employee.getSalaryPerHour();
        }
        return Integer.parseInt(String.valueOf(total));
    }

    // overridden toString method
    @Override
```

```
    public String toString() {
        String employeeString = "";
        // sort the arraylist using compareTo method defined in Employee class
        Collections.sort(listOfEmployees);
        // loop through array list

        for (int i = 0; i < listOfEmployees.size(); i++) {
            // get employee at index i
            Employee employee = listOfEmployees.get(i);
            employeeString += employee.toString();
        }
        return employeeString;
    }
}
```

ProjectPrinter.java

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;

public class ProjectPrinter implements Runnable {
    private Project project;

    // getter method
    public Project getProject() {
        return project;
    }

    // constructor
    public ProjectPrinter(Project project) {
        this.project = project;
    }

    // setter method
    public void setProject(Project project) {
        this.project = project;
    }

    // implements the run method of Runnable interface
    @Override
    public void run() {

        // get arraylist of employees
        ArrayList<Employee> listOfEmployees = project.getListOfEmployees();
        try {
            // initialize date formatter
            SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
            // get formatted dates
            String sDate = formatter.format(project.getStartDate());
            String eDate = formatter.format(project.getEndDate());
            String projectStr = "[Project Id:" + project.getProjectId() + " - " + "Project Duration::" + sDate + " to "
                    + eDate + "]";
            System.out.println(projectStr);

            // loop through employees array list
            for (int i = 0; i < listOfEmployees.size(); i++) {
                // get employee at index i
                Employee employee = listOfEmployees.get(i);
```

```
                // print employee
                System.out.println(employee);
                // delay of 1 second
                Thread.sleep(1000);
            } // catch InterruptedException
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}
```