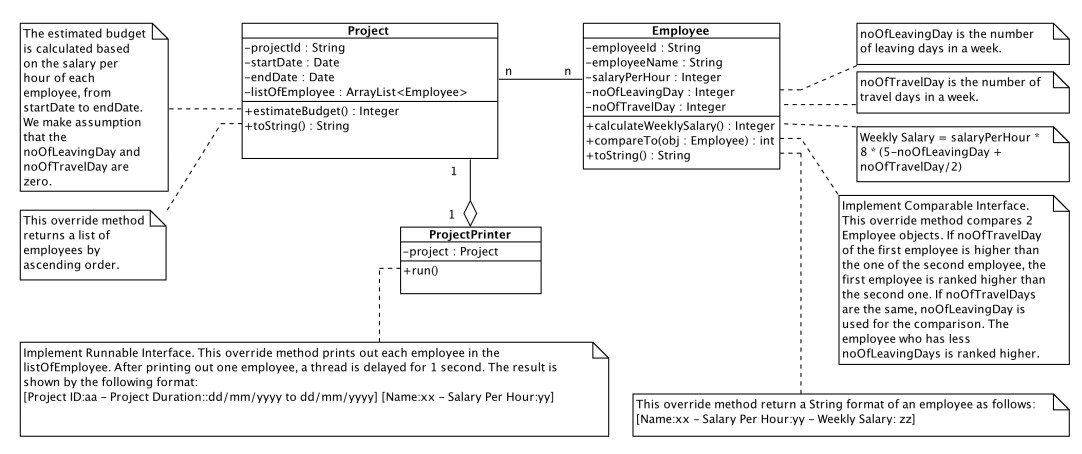
**Object-Oriented Programming**

**Lab Session #5**

You are required to implement the following design as well as a **Main** class to test your implementation. You must add appropriate constructors, getter and setter for each class. You can add private methods into the classes to support your work.

import java.util.ArrayList;

import java.util.Collections;

import java.util.Date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

public class Project {

private String projectid;

private Date startdate;

private Date enddate;

private ArrayList<Employee> listofemployee;

public String getProjectid() {

return projectid;

}

public void setProjectId(String projectid) {

this.projectid = projectid;

}

public String getStartdate() {

return new SimpleDateFormat("dd/mm/yyyy").format(startdate);

}

public void setStart(Date date) {

startdate = date;

}

public String getEnddate() {

return new SimpleDateFormat("dd/mm/yyyy").format(enddate);

}

public void setEnddate(Date date) {

this.enddate = date;

}

public ArrayList<Employee> getListofemployee() {

return listofemployee; }

public Project(String projectid, String start, String end) {

SimpleDateFormat format = new SimpleDateFormat("dd/mm/yyyy");

this.projectid = projectid;

try {

this.startdate = format.parse(start);

this.enddate = format.parse(end);

} catch (ParseException e) {

e.printStackTrace();

}

listofemployee = new ArrayList<Employee>();

}

public void addEmployee(Employee em) {

listofemployee.add(em);

}

public int estimateBudget() {

int budget = 0;

long er = enddate.getTime() - startdate.getTime();

int workingDays = (int)er / (1000 \* 3600 \* 24);

System.***out***.println(workingDays);

for (Employee e : listofemployee) {

budget += workingDays \* e.getSalaryPerHour() \* 8;

}

return budget;

}

public String toString() {

Collections.*sort*(listofemployee);

return listofemployee.toString();

}

}

public class Employee implements Comparable<Employee>{

private String employeeid;

private String employeeName;

private int salaryPerHour;

private int noOfLeavingDay;

private int noOfTravelDay;

public Employee(Project project, String employeeid, String employeeName, int salaryPerHour, int noOfLeavingDay, int noOfTravelDay) {

project.addEmployee(this);

this.setEmployeeid(employeeid);

this.setEmployeeName(employeeName);

this.setSalaryPerHour(salaryPerHour);

this.setNoOfLeavingDay(noOfLeavingDay);

this.setNoOfTravelDay(noOfTravelDay);

}

public String getEmployeeid() {

return employeeid;

}

public void setEmployeeid(String employeeid) {

this.employeeid = employeeid;

}

public String getEmployeeName() {

return employeeName;

}

public void setEmployeeName(String employeeName) {

this.employeeName = employeeName;

}

public int getNoOfLeavingDay() {

return noOfLeavingDay;

}

public void setNoOfLeavingDay(int noOfLeavingDay) {

this.noOfLeavingDay = noOfLeavingDay;

}

public int getSalaryPerHour() {

return salaryPerHour;

}

public void setSalaryPerHour(int salaryPerHour) {

this.salaryPerHour = salaryPerHour;

}

public int getNoOfTravelDay() {

return noOfTravelDay;

}

public void setNoOfTravelDay(int noOfTravelDay) {

this.noOfTravelDay = noOfTravelDay;

}

public int calculateWeeklysalary() {

return salaryPerHour\*8\*(5-noOfLeavingDay + noOfTravelDay/2);

}

*@Override*

public int compareTo(Employee e) {

if (this.noOfTravelDay > e.noOfTravelDay)

return 1;

else if (this.noOfTravelDay < e.noOfTravelDay)

return -1;

else {

if (this.noOfLeavingDay < e.noOfLeavingDay)

return 1;

else if (this.noOfLeavingDay > e.noOfLeavingDay)

return -1;

else

return 0;

}

}

*@Override*

public String toString() {

return "Name: " + employeeName + "- Salary Per Hour: " + salaryPerHour + "- Weekly Salary: " + calculateWeeklysalary();

}

}

public class ProjectPrinter implements Runnable{

private Project project;

public ProjectPrinter(Project project) {

this.project = project;

}

public void run() {

for (Employee em : project.getListofemployee()) {

System.***out***.println("[" + "Project ID: " + project.getProjectid() + " - Project Duration: " + project.getStartdate() + " to " + project.getEnddate() + "] [Name: " + em.getEmployeeName() + " - Salary Per Hour: " + em.getSalaryPerHour() + "]");

try {

Thread.*sleep*(2000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

public class Main {

public static void main(String[] args) {

Project project = new Project("ITDSIU19023", "26/05/2020", "26/12/2020");

ProjectPrinter print = new ProjectPrinter(project);

Employee em1 = new Employee(project, "ITDS", "TrungLy", 100, 0, 0);

Employee em2 = new Employee(project, "ITDS2", "MinhTrung", 200, 1, 3);

Employee em3 = new Employee(project, "ITDS3", "LyTrung", 50, 2, 2);

Thread thread = new Thread(print);

thread.start();

}

}