UC1

public class EmpWageBuilderUC1

{

Public static void main(String []args)

{

Int IS\_FULL\_TIME=1;

Double empCheck =Math.floor(Math.random() \* 10 )%2;

If(empcheck == IS\_FULL\_TIME) {

System.out.println(“Employee is Present”);

}

Else

{

System.out.println(“Employee is Absent”);

}

UC2

Public class EmpWageBuilderUC1

{

Public static void main(String []args)

{

Int IS \_FULL\_TIME=1;

Int Emp\_Rate\_PER\_Hour=20;

Int empHrs=0;

Int empWage=0;

Double empcheck=Math.floor(Math.random() \*10) %2;

If(empCheck == IS\_FULL\_TIME)

empHrs=8;

else

empHrs=0;

empWage =empHrs \*EMP\_RATE\_PER\_HOUR;

System.out.println(“EmpWage:+empWage”);

}

}

UC3

Public class EmpWageBuilderUC3

{

Public static void main(String []args)

{

Int IS \_Part\_TIME=1;

Int IS \_FULL\_TIME=2;

Int Emp\_Rate\_PER\_Hour=20;

Int empHrs=0;

Int empWage=0;

Double empcheck=Math.floor(Math.random() \*10) %3;

If(empCheck == IS \_Part\_TIME)

empHrs=4;

else if If(empCheck == IS \_Full\_TIME)

empHrs=8;

else

empHrs=0;

empWage =empHrs \*EMP\_RATE\_PER\_HOUR;

System.out.println(“EmpWage:+empWage”);

}

}

UC\_4

Public class EmployeeWageBuilderMultiCompany

{

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

**Public static final**  **int** EMP\_RATE\_PER\_HOUR=1;

Int empHrs=0;

Int empWage=0;

Double empcheck=Math.floor(Math.random() \*10) %3;

If(empCheck == IS \_Part\_TIME)

empHrs=4;

break;

If(empCheck == IS \_FULL\_TIME)

empHrs=8;

break;

default:

empHrs=0;

}

empWage=empHrs \* EMP\_RATE\_PER\_HOUR:

System.out.println(Emp Wage:”+ empWage);

}

}

UC\_5

Public class EmployeeWageBuilderMultiCompany

{

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

**Public static final**  **int** EMP\_RATE\_PER\_HOUR=20;

**Public static final**  **int** NUM\_OF\_WORKING\_DAYS=2;

int empHrs=0;

Int empWage=0;

totalEmpWage=0;

for(int day =0;day<NUM\_OF\_WORKING\_DAYS;day++)

int empcheck=(int)Math.floor(Math.random() \*10) %3;

switch(empCheck )

case IS \_Part\_TIME;

empHrs=4;

break;

If(empCheck == IS \_FULL\_TIME)

empHrs=8;

break;

default:

empHrs=0;

}

empWage=empHrs \* EMP\_RATE\_PER\_HOUR:

System.out.println(“Emp Wage:”+ empWage);

}

System.out.println(“Total Emp Wage:”+total empWage);

}

UC\_6

public class EmployeeWageBuilderMultiCompany

{

Int empHrs=0;

Int empWage=0;

totalEmpWage=0;

while(totalEmpHrs=<=MAX\_HRS\_IN\_MONTH&&

totalWorkingDays<NUM\_OF\_WORKING\_DAYS)

totalWorkingDays++;

int empcheck=(int)Math.floor(Math.random() \*10) %3;

switch(empCheck )

case IS \_Part\_TIME;

empHrs=4;

break;

If(empCheck == IS \_FULL\_TIME)

empHrs=8;

break;

default:

empHrs=0;

}

empHrs=empHrs ;

System.out.println(“totalWorkingDAys:”+ totalWorkingDAys);

}

Int totalEmpWage=totalEmpHrs \* EMP\_RATE\_PER\_HOUR;

System.out.println(“Total Emp Wage:”+total empWage);

}

UC\_7

Public class EmployeeWageBuilderClass

{

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

**Public static final**  **int** EMP\_RATE\_PER\_HOUR=1;

**Public static final**  **int** NUM\_OF\_WORKING\_DAYS=2;

**Public static final**  **int** MAX\_HOURS\_IN\_MONTH=10;

public static int ComputeWage()

int empHrs=0;

total EmpHrs=0;

totalworkingDays=0;

while(totalEmpHrs<=maxHours\_IN\_MONTH && totalworkingDays<NUM\_of\_working\_Days){

totalWorkingDays++;

int empCheck=(int) Math.floor(math.random() \* 10)%3;

switch(empCheck)

{

Case Is\_PART\_TIME;

empHrs=4;

break;

Case Is\_FULL\_TIME;

empHrs=8;

break;

default:

empHrs=0;

}

totalEmpHrs += empHrs;

System.out.println(“DAY#”total working days +”EMP hr”+empHrs);

}\_

Int totalEmpWage=totalEmpHrs \* EMP\_RATE\_Per\_HOUR;

System.out.println(“total Emp wage for company: ”+company+” is:”+totalEmpWage);

return totalEmpWage;

}

Public static void main (String[]args)

{

computeEmpWage();

}

}

UC\_8

Public class EmployeeWageBuilderMultiCompany

{

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

public static int ComputeWage(String company, int empRatePerhour,int noOfworking days,int maxHoursperMonth)

int empHrs=0;

total EmpHrs=0;

totalworkingDays=0;

while(totalEmpHrs<=maxHoursPerMonth && totalworkingDays<no.of working days){

totalWorkingDays++;

int empCheck=(int) Math.floor(math.random() \* 10)%3;

switch(empCheck)

{

Case Is\_PART\_TIME;

empHrs=4;

break;

Case Is\_FULL\_TIME;

empHrs=8;

break;

default:

empHrs=0;

}

totalEmpHrs +=empHrs;

System.out.println(“DAY#”total working days +”EMP hr”+empHrs);

}

Int totalEmpWage=totalEmpHrs \* empRatePerHr;

System.out.println(“total Emp wage for company: ”+company+” is:”+totalEmpWage);

return totalEmpWage;

}

Public static void main (String[]args)

{

computeEmpWage(“DMART”,20,2,10);

computeEmpWage(“Realiance”,20,2,10);

}

}

UC\_9

Public class EmployeeBuilderOBJECT

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

private final string company;

private final int empRatePerHour;

private final int noofWorkingDays;

private final int maxHoursPerMonth;

private final int EmpWage;

public EmpWageBuilderObject(String company int empRatePerHour,int numOfWorkingDays,int MaxHoursPerMonth)

{

this.company=company;

this.empRatePerHour;

this.no.ofWorkingDays=numOfWorkingDays;

this.maxHoursPerMonth=maxHoursPerMonth;

}

**Public void ComputeWage**

int empHrs=0;

total EmpHrs=0;

totalworkingDays=0;

while(totalEmpHrs<=maxHoursPerMonth && totalworkingDays<no.of working days){

totalWorkingDays++;

int empCheck=(int) Math.floor(math.random() \* 10)%3;

switch(empCheck)

{

Case Is\_PART\_TIME;

empHrs=4;

break;

Case Is\_FULL\_TIME;

empHrs=8;

break;

default:

empHrs=0;

}

totalEmpHrs +=empHrs;

System.out.println(“DAY#”total working days +”EMP hr”+empHrs);

}

totalEmpWage=totalEmpHrs \* empRatePerHour;

}

Public StringtoString()

{

return “totalemp wage for company+”+total EmpWage;

}

System.out.println(“total Emp wage for company: ”+company+” is:”+totalEmpWage);

return totalEmpWage;

}

Public static void main (String[]args)

{

EmpWageBuilderObject dMart =new EmpWageBuilderObject (“DMART”,20,2,10);

EmpWageBuilderObject realiance=new EmpWageBuilderObject (“Realiance”,20,2,10);

Dmart.computeEmpWage();

System.out.println(“dMart”);

realince.computeEmpWage()

System.out.println(“realince”);

}

}

UC10

Public class EmployeeBuilderOBJECT

public final string company;

public final int empRatePerHour;

public final int noofWorkingDays;

public final int maxHoursPerMonth;

public final int EmpWage;

public EmpWageBuilderObject(String company int empRatePerHour,int numOfWorkingDays,int MaxHoursPerMonth)

{

this.company=company;

this.empRatePerHour;

this.no.ofWorkingDays=numOfWorkingDays;

this.maxHoursPerMonth=maxHoursPerMonth;

}

Public StringtoString()

{

return “totalemp wage for company+”+total EmpWage;

}

**Public static final**  **int** IS\_PART\_TIME=1;

**Public static final** **int** IS\_FULL\_TIME=2;

private int numOfCompany=0;

private CompanyEmpWage[] companyEmpWageArray;

public EmpWageBuilderArray()

{

companyEmpWageArray new CompanyEmpWage[5];

}

Private void addCompanyEmpWage(String company,int empRatePerHour,int numOfWorkingDays,int maxHoursPerMonth)

{

CompanyEmpWageArray(numOfCompany)=new CompanyEmpWage(company,empRatePerHour,numOfWorkingDays,maxHorsPerMonth);

Private void computeEmpWage()

{

For(int i=0;i< numOfCompany;i++)

{

companyEmpWageArray[i].setTotalEmpWage(this.computeEmpWage(companyEmpWageArray)));

System.out.println(CompanyEmpWageArray[i]);

}

}

switch(empCheck)

{

Case Is\_PART\_TIME;

empHrs=4;

break;

Case Is\_FULL\_TIME;

empHrs=8;

break;

default:

empHrs=0;

}

totalEmpHours= += empHrs;

System.out.println(“Day: ”+ totalWorkingDay +” empHr:”+empHrs);

return totalEmpHrs \* computeEmpWage.empRatePerHour;

Public static void main (String[]args)

{

EmpBuilderArray empBuilderArray=new EmpBuilderArray();

EmpWageBuilderObject dMart =new EmpWageBuilderObject (“DMART”,20,2,10);

EmpWageBuilderObject realiance=new EmpWageBuilderObject (“Realiance”,20,2,10);

empWageBuilder.computeEmpWage();

}

}

UC11

|  |
| --- |
| Package.com.collection; |
|  |  |  |
|  |  | import java.util.\*; |
|  |  |  |
|  |  | public class EmpWageBuilder implements IComputeEmpWage { |
|  |  | public static final int IS\_PART\_TIME = 1; |
|  |  | public static final int IS\_FULL\_TIME = 2; |
|  |  |  |
|  |  | public ArrayList<CompanyEmpWage> companyList = new ArrayList<CompanyEmpWage>(); |
|  |  | public Map<String, CompanyEmpWage> companyToEmpWageMap = new HashMap<String, CompanyEmpWage>(); |
|  |  |  |
|  |  | public void computeEmpWage() { |
|  |  | for (int i = 0; i < companyList.size(); i++) { |
|  | | @@ -43,10 +44,25 @@ public static void computeEmpWage(CompanyEmpWage companyEmpWage) { |
|  |  |  |
|  |  | } |
|  |  |  |
|  |  | public void addEmpWageMap() { |
|  |  | for (int i = 0; i < companyList.size(); i++) { |
|  |  | companyToEmpWageMap.put(companyList.get(i).company, companyList.get(i)); |
|  |  | } |
|  |  | } |
|  |  |  |
|  |  | public void displayMap() { |
|  |  | for (Map.Entry<String, CompanyEmpWage> entry : companyToEmpWageMap.entrySet()) { |
|  |  | System.out.println("Company: " + entry.getKey() + " Total Wage:" + entry.getValue().totalEmpWage); |
|  |  | } |
|  |  | } |
|  |  |  |
|  |  | public static void main(String[] args) { |
|  |  | EmpWageBuilder emp = new EmpWageBuilder(); |
|  |  | emp.companyList.add(new CompanyEmpWage("Infosys", 20, 2, 10)); |
|  |  | emp.companyList.add(new CompanyEmpWage("Syntel", 10, 4, 20)); |
|  |  | emp.computeEmpWage(); |
|  |  | emp.addEmpWageMap(); |
|  |  | emp.displayMap(); |
|  |  | System.out.println(emp.companyToEmpWageMap.get("Infosys").totalEmpWage); |
|  |  | } |
|  |  | } |

UC12

|  |
| --- |
| package com.collectiond |
|  |  |  |
|  |  | **import java.util.\*;** |
|  |  |  |
|  |  | public class EmpWageBuilder implements IComputeEmpWage { |
|  |  | public static final int IS\_PART\_TIME = 1; |
|  |  | public static final int IS\_FULL\_TIME = 2; |
|  |  |  |
|  |  | public ArrayList<CompanyEmpWage> companyList = new ArrayList<CompanyEmpWage>(); |
|  |  | public Map<String, CompanyEmpWage> companyToEmpWageMap = new HashMap<String, CompanyEmpWage>(); |
|  |  |  |
|  |  | public void computeEmpWage() { |
|  |  | for (int i = 0; i < companyList.size(); i++) { |
|  | | @@ -43,10 +44,25 @@ public static void computeEmpWage(CompanyEmpWage companyEmpWage) { |
|  |  |  |
|  |  | } |
|  |  |  |
|  |  | public void addEmpWageMap() { |
|  |  | for (int i = 0; i < companyList.size(); i++) { |
|  |  | companyToEmpWageMap.put(companyList.get(i).company, companyList.get(i)); |
|  |  | } |
|  |  | } |
|  |  |  |
|  |  | public void displayMap() { |
|  |  | for (Map.Entry<String, CompanyEmpWage> entry : companyToEmpWageMap.entrySet()) { |
|  |  | System.out.println("Company: " + entry.getKey() + " Total Wage:" + entry.getValue().totalEmpWage); |
|  |  | } |
|  |  | } |
|  |  |  |
|  |  | public static void main(String[] args) { |
|  |  | EmpWageBuilder emp = new EmpWageBuilder(); |
|  |  | emp.companyList.add(new CompanyEmpWage("Infosys", 20, 2, 10)); |
|  |  | emp.companyList.add(new CompanyEmpWage("Reliance", 10, 4, 20)); |
|  |  | emp.computeEmpWage(); |
|  |  | emp.addEmpWageMap(); |
|  |  | emp.displayMap(); |
|  |  | System.out.println(emp.companyToEmpWageMap.get("Infosys").totalEmpWage); |
|  |  | System.out.println(emp.companyToEmpWageMap.get("Realince").totalEmpWage); |
|  |  | } |

UC13

package com.collection;

import java.util.\*;

public class EmpWageBuilder implements IComputeEmpWage {

public static final int IS\_PART\_TIME = 1;

public static final int IS\_FULL\_TIME = 2;

public ArrayList<CompanyEmpWage> companyList = new ArrayList<CompanyEmpWage>();

public Map<String, CompanyEmpWage> companyToEmpWageMap = new HashMap<String, CompanyEmpWage>();

public void computeEmpWage() {

for (int i = 0; i < companyList.size(); i++) {

public static void computeEmpWage(CompanyEmpWage companyEmpWage) {

public void addEmpWageMap() {

for (int i = 0; i < companyList.size(); i++) {

companyToEmpWageMap.put(companyList.get(i).company, companyList.get(i));

}

}

public void displayMap() {

for (Map.Entry<String, CompanyEmpWage> entry : companyToEmpWageMap.entrySet()) {

System.out.println("Company: " + entry.getKey() + " Total Wage:" + entry.getValue().totalEmpWage);

}

}

public static void main(String[] args) {

EmpWageBuilder emp = new EmpWageBuilder();

emp.companyList.add(new CompanyEmpWage("DMart", 20, 2, 10));

emp.companyList.add(new CompanyEmpWage("Reliance", 10, 4, 20));

emp.computeEmpWage();

emp.addEmpWageMap();

emp.displayMap();

System.out.println(emp.companyToEmpWageMap.get("DMart").totalEmpWage);

}

}