/\*\* =======================================================================

\* Class:Worker ExT.T Pg.N/A Author: Yin Linhai

\* Version:001Date:Apr 10, 2014

\*

\* A program which helps a company manager keep track of wages

\*

\* Course:Computer Science 202Teacher:Mr Blakey

\* School:Sir Winston Churchill High School, Calgary, Alberta, Canada

\* Language: Java SE 7.0Target Operating System: Java Virtual Machine

\* System:Intel Celeron 3GHz running under Windows 7 IDE: Eclipse 4.2

\*========================================================================\*/

**Worker Class**

**package** test\_9;

**public** **class** Worker {

//constructor

**public** Worker(String named, **float** wages) {

name = named;

setWage(wages);

}

//methods

//compute pay

**public** **void** computePay(**int** hour) {

setHours(hour);

setPayEarned(getWage()\*getHours());

}

//get setters

**public** String getName() {

**return** name;

}

**public** **int** getHours() {

**return** hours;

}

**public** **float** getPay() {

**return** getPayEarned();

}

**public** **float** getWage() {

**return** wage;

}

**public** **void** setWage(**float** wage) {

**this**.wage = wage;

}

**public** **float** getPayEarned() {

**return** payEarned;

}

**public** **void** setPayEarned(**float** payEarned) {

**this**.payEarned = payEarned;

}

**public** **void** setHours(**int** hours) {

**this**.hours = hours;

}

//state variables

**public** String name;

**private** **int** hours;

**private** **float** payEarned, wage;

}

**HourlyWorker Class**

**package** test\_9;

**public** **class** HourlyWorker **extends** Worker {

//constructor

**public** HourlyWorker(String name, **float** wage) {

**super**(name,wage);

}

//compute pay method

**public** **void** computePay(**int** hour) {

setHours(hour);

**if**(getHours()>40) {

setPayEarned(getWage()\*40 + (getHours()-40)\*getWage()/2);

} **else** {

setPayEarned(getHours()\*getWage());

}

}

}

**SalariedWorker Class**

**package** test\_9;

**public** **class** SalariedWorker **extends** Worker {

//constructor

**public** SalariedWorker(String name, **float** wage) {

**super**(name, wage);

}

//compute pay method

**public** **void** computePay(**int** hour) {

setHours(hour);

setPayEarned(getWage()\*40);

}

}

**Main Class**

**package** test\_9;

**import** java.util.ArrayList;

**import** java.util.Scanner;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

//variables

**boolean** quit = **false**;

//object constructor

Scanner scan = **new** Scanner(System.*in*);

Scanner scan1 = **new** Scanner(System.*in*);

ArrayList<Worker> list = **new** ArrayList<Worker>();

**while**(!quit) {

//print out menu

System.*out*.println("What would you like to do?");

System.*out*.println("1. Add workers\n2. Change worked hours");

System.*out*.println("3. Print out wages to be paid");

System.*out*.println("4. Remove workers\n5. Quit");

//try block in case of errors

**try** {

//switch for menu

**switch**(scan.nextInt()){

//Block which creates workers

//ask for name of new worker

**case** 1: System.*out*.println("\nWhat is the name of the person you would like to add?");

String name = scan1.nextLine();

//ask for wage

System.*out*.println("\nWhat is their wage?\n($/h)");

**float** wage = scan.nextFloat();

//block which asks for how they're paid

System.*out*.println("\nAre they paid hourly?\nY/N");

String type = scan.next();

//if its an hourly worker

**if**(type.equalsIgnoreCase("Y")) {

HourlyWorker a = **new** HourlyWorker(name, wage);

list.add(a);

} **else** {

//ask for confirmation if they are a salary worker

System.*out*.println("\nAre they on a salary?\nY/N");

String type2 = scan.next();

**if**(type2.equalsIgnoreCase("Y")) {

//create salary worker, and add pay

SalariedWorker s = **new** SalariedWorker(name, wage);

s.computePay(0);

list.add(s);

} **else** {

System.*out*.println("Cannot make that worker");

}

}

**break**;

//block for adding hours worked

**case** 2: System.*out*.println("\nTo all or just one?\n(one/all)");

String all = scan.next();

//to go through all workers

**if**(all.equalsIgnoreCase("all")) {

//add hours, and compute pay

**for**(**int** x = 0; x<list.size(); x++) {

Worker temp = list.get(x);

System.*out*.println("\nHow many hours has " + temp.getName() + " worked?");

temp.computePay(scan.nextInt());

}

} **else** {

//to find one

System.*out*.println("\nWhat is the name of the worker?");

String nameCheck = scan1.nextLine();

**for**(**int** x = 0; x<list.size(); x++) {

Worker temp = list.get(x);

//check name

**if**(temp.getName().equalsIgnoreCase(nameCheck)) {

//add hours, and compute pay

System.*out*.println("\nHow many hours have they worked?");

temp.computePay(scan.nextInt());

**break**;

}

}

}

**break**;

//block which prints out data

**case** 3: System.*out*.println("Name\tHours\tWage");

**float** total = 0;

**for**(**int** x = 0; x<list.size(); x++) {

Worker temp = list.get(x);

System.*out*.print("\n" + temp.getName()+"\t");

System.*out*.print(temp.getHours()+"\t");

System.*out*.print(String.*format*("%9.2f",temp.getPayEarned()));

System.*out*.println();

total += temp.getPayEarned();

}

System.*out*.println("\nThe company has to pay $" + String.*format*("%9.2f", total) + ".\n");

**break**;

//block which removes workers

//asks for worker name

**case** 4: System.*out*.println("\nWhat is the name of the worker?");

String nameCheck = scan1.nextLine();

//search array

**for**(**int** x = 0; x<list.size(); x++) {

Worker temp = list.get(x);

//check name

**if**(temp.getName().equalsIgnoreCase(nameCheck)) {

//remove worker

list.remove(x);

System.*out*.println("\nThe worker has succesfully been removed\n");

}

}

**break**;

//quit block

**case** 5: quit = **true**;

**break**;

//default

**default**:System.*out*.println("Thats not a selection!");

}

}

//catch runtime error and report

**catch**(RuntimeException e) {

String error = scan.nextLine();

System.*out*.println("The program cannot interpret " + error + ".\n");

}

}

scan.close();

scan1.close();

}

}

Output:

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

1

What is the name of the person you would like to add?

bob

What is their wage?

($/h)

5

Are they paid hourly?

Y/N

y

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

1

What is the name of the person you would like to add?

Jazz

What is their wage?

($/h)

20

Are they paid hourly?

Y/N

n

Are they on a salary?

Y/N

y

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

1

What is the name of the person you would like to add?

test

What is their wage?

($/h)

99.99

Are they paid hourly?

Y/N

y

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

2

To all or just one?

(one/all)

all

How many hours has bob worked?

10

How many hours has Jazz worked?

54

How many hours has test worked?

32

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

3

Name Hours Wage

bob 10 50.00

Jazz 54 800.00

test 32 3199.68

The company has to pay $ 4049.68.

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

4

What is the name of the worker?

test

The worker has succesfully been removed

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

3

Name Hours Wage

bob 10 50.00

Jazz 54 800.00

The company has to pay $ 850.00.

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

1

What is the name of the person you would like to add?

134

What is their wage?

($/h)

sdas

The program cannot interpret sdas.

What would you like to do?

1. Add workers

2. Change worked hours

3. Print out wages to be paid

4. Remove workers

5. Quit

5