Michelle Zulli

Dr. Alireza Ebrahimi

Introduction to C++ and OOP

August 16, 2015

# Module 6: Payroll

The program was modified to use two derived classes (hourly and salary). Virtual functions were defined in the parent class (employee) for calculating hours and grosspay. The employees’ data was calculated and all employee objects were placed in arrays. A single pointer array was created for purposes of sorting by netpay.

## Code

#include<iostream>

#include<fstream>

#include<iomanip>

using namespace std;

class employee {

public:

ifstream fin;

string firstname, lastname;

int id;

float hours, base\_hours, ot\_hours, rate, ot\_rate, grosspay, netpay;

float taxamount, taxrate;

char status;

// set base and ot hours

void findhours() {

base\_hours = hours;

ot\_hours = 0;

if (base\_hours > 40) {

ot\_hours = base\_hours - 40;

base\_hours = 40;

}

} // findhours

// set ot rate

virtual void findotrate() {}

// calculate gross pay

virtual void findgrosspay() {}

// calculate tax amount

void findtaxamt() {

taxamount = grosspay \* taxrate;

} // findtaxamt

// calculate net pay

void findnetpay() {

netpay = grosspay - taxamount;

} // findnetpay

// print table headers

void printheader() {

cout << setiosflags(ios::left)

<< " ZULLI PAYROLL"

<< endl

<< setw(16) << "FIRST NAME" << setw(16) << "LAST NAME"

<< setw(8) << "STATUS" << setw(6) << "ID" << setw(8) << "HOURS"

<< setw(10) << "OT HOURS" << setw(10) << "RATE"

<< setw(10) << "OT RATE" << setw(10) << "GROSS"

<< setw(10) << "TAX" << setw(10) << "NET"

<< endl

<< "=============== =============== ======= ===== ======= "

<< "========= ========= ========= ========= ========= ========="

<< endl;

} // printheader

// print data for employee

void printdata() {

cout << setprecision(2)

<< setiosflags(ios::showpoint | ios::fixed | ios::left)

<< endl

<< setw(16) << firstname << setw(16) << lastname

<< setw(8) << status << setw(6) << id

<< setw(8) << hours << setw(10) << ot\_hours << setw(10) << rate

<< setw(10) << ot\_rate << setw(10) << grosspay

<< setw(10) << taxrate << setw(10) << netpay

<< endl;

} // printdata

employee() {

taxrate = .3;

}

~employee() {

}

};

class hourly: public employee {

public:

// set ot rate

void findotrate() {

ot\_rate = rate \* 1.5;

} // findotrate

// calculate gross pay

void findgrosspay() {

grosspay = (base\_hours \* rate) + (ot\_hours \* ot\_rate);

} // findgrosspay

hourly() {

fin.open("hourly.in");

}

~hourly() {

fin.close();

}

};

class salary: public employee {

public:

// set ot rate

void findotrate() {

ot\_rate = rate / 52 / 40 \* 1.5;

} // findotrate

// calculate gross pay

void findgrosspay() {

grosspay = rate / 52 + (ot\_hours \* ot\_rate);

} // findgrosspay

salary() {

fin.open("salary.in");

}

~salary() {

fin.close();

}

};

int main() {

employee empl;

// print table header

empl.printheader();

// create array of hourly employees

hourly hrly;

hourly hEmp[3];

for (int i = 0; i < 3; i++) {

hrly.fin >> hEmp[i].firstname >> hEmp[i].lastname >> hEmp[i].status >> hEmp[i].id >> hEmp[i].hours >> hEmp[i].rate;

hEmp[i].findhours();

hEmp[i].findotrate();

hEmp[i].findgrosspay();

hEmp[i].findtaxamt();

hEmp[i].findgrosspay();

hEmp[i].findnetpay();

}

// create array of salary employees

salary slry;

salary sEmp[3];

for (int i = 0; i < 3; i++) {

slry.fin >> sEmp[i].firstname >> sEmp[i].lastname >> sEmp[i].status >> sEmp[i].id >> sEmp[i].hours >> sEmp[i].rate;

sEmp[i].findhours();

sEmp[i].findotrate();

sEmp[i].findgrosspay();

sEmp[i].findtaxamt();

sEmp[i].findgrosspay();

sEmp[i].findnetpay();

}

// fill pointer array with hourly and salary employees

employee \*staff[6];

for (int i = 0; i < 3; i++) {

staff[i] = hEmp+i;

}

int j = 0;

for (int i = 3; i < 6; i++) {

staff[i] = sEmp+j;

j++;

}

// sort pointer array

for (int i = 0; i < 5; i++) {

for (int j = 5; j > i; j--) {

if (staff[j]->netpay < staff[j-1]->netpay) {

employee \*temp = staff[j];

staff[j] = staff[j-1];

staff[j-1] = temp;

}

}

}

// print pointer array

for (int i = 0; i < 6; i++) {

staff[i]->printdata();

}

// print highest and lowest net

cout << endl;

cout << "LOWEST NET : $ " << staff[0]->netpay << " (" << staff[0]->firstname << " " << staff[0]->lastname << ")" << endl;

cout << "HIGHEST NET: $" << staff[5]->netpay << " (" << staff[5]->firstname << " " << staff[5]->lastname << ")" << endl;

} // MAIN

## Output

