**Screen Shot:**

I was unable to make my code work. I need help.

**Code:**

// ---------------------------------------------------------------------------

//Program Header

//Program Name: Employee Class Integration

//Programmer: Renee Thomas

//CIS247, Week 5 Lab

//Program Description: Update Employee Class to inclued a Salaryied and Hourly employees

// ---------------------------------------------------------------------------

# include <iostream>

# include <string>

# include <iomanip>

using namespace std;

// create prototypes for functions

void DisplayApplicationInformation();

void DisplayDivider(string);

string getInput(string);

void TerminateApplication();

// create Benefits Class

class Benefits{

private:

string healthInsurance;

double lifeInsurance;

int vacation;

public:

Benefits();

Benefits(string hI, double lI, int vac);

void displayBenefits();

string getHealthInsurance();

void setHealthInsurance(string hI);

double getLifeInsurance();

void setLifeInsurance(double lI);

int getVacation();

void setVacation(int vac);

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Benefits Class Functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Benefits::Benefits(){

healthInsurance = "Not provided";

lifeInsurance = 0.0;

vacation = 14;

}

Benefits::Benefits(string hI, double lI, int vac){

healthInsurance = hI;

lifeInsurance = lI;

vacation = vac;

}

void Benefits::displayBenefits(){

cout<<"Heath Insurance: "<<healthInsurance<<endl;

cout<<"Life Insurance: "<<lifeInsurance<<endl;

cout<<"Vacation: "<<vacation<<endl;

}

string Benefits::getHealthInsurance(){return healthInsurance;}

void Benefits::setHealthInsurance(string hI){healthInsurance = hI;}

double Benefits::getLifeInsurance(){return lifeInsurance;}

void Benefits::setLifeInsurance(double lI){lifeInsurance = lI;}

int Benefits::getVacation(){return vacation;}

void Benefits::setVacation(int vac){vacation = vac;}

// create Employee Class

class Employee

{

private:

static int numEmployees;

protected:

string firstName;

string lastName;

char gender;

int dependents;

double annualSalary;

Benefits bens;// create an instance of Benefits inside Employee class

public:

Employee();

Employee(string fn, string ln, char gen, int dep, double sal, Benefits bens\_in); // don't forget to use the class name as the data type!

double calculatePay(double sal);

void displayEmployee();

string getFirstName();

void setFirstName(string fn);

string getLastName();

void setLastName(string ln);

char getGender();

void setGender(char gen);

int getDependents();

void setDependents(int dep);

double getAnnualSalary();

void setAnnualSalary(double sal);

static int getNumEmployees();

void setDependents(string dep);

void setAnnualSalary(string sal);

void setBenefits(Benefits);

Benefits getBenefits();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Employee Class Functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int Employee::numEmployees = 0;

// default constructor

Employee::Employee()

{

firstName = "Not Given";

lastName = "Not Given";

gender = 'U';

dependents = 0;

annualSalary = 20000;

bens = Benefits();

numEmployees ++;

}

// Employee constructor with multi arguements (remember to use class name as the data type when integrating into parameters)

Employee::Employee(string fn, string ln, char gen, int dep, double sal, Benefits bens\_in )

{

firstName = fn;

lastName = ln;

gender = gen;

dependents = dep;

annualSalary = sal;

bens = bens\_in;// set original instance to parameter instance

numEmployees ++;

}

// create function for calculatePay

double Employee:: calculatePay(double sal){

double wsal = sal/52;

return wsal;

}

// create function to display all employee information

void Employee::displayEmployee()

{

cout<<"\nEmployee Information"<<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl<<endl;

cout<<"Employee Name: \t"<<firstName<<" "<<lastName<<endl;

cout<<"Gender: \t"<<gender<<endl;

cout<<"Dependents: \t"<<dependents<<endl;

cout<<"Annual Salary: \t"<< setprecision(2) << showpoint << fixed << annualSalary<<endl;

cout<<"Weekly Salary: \t"<< setprecision(2) << showpoint << fixed <<calculatePay(annualSalary)<<endl;

bens.displayBenefits();

cout<<endl<<endl;

}

// create getters and setters for all private variables

string Employee::getFirstName(){return firstName;}

void Employee::setFirstName(string fn){firstName=fn;}

string Employee::getLastName(){return lastName;}

void Employee::setLastName(string ln){lastName=ln;}

char Employee::getGender(){return gender;}

void Employee::setGender(char gen){gender=gen;}

int Employee::getDependents(){return dependents;}

void Employee::setDependents(int dep){dependents=dep;}

void Employee::setDependents(string dep){dependents= atoi(dep.c\_str());}

double Employee::getAnnualSalary(){return annualSalary;}

void Employee::setAnnualSalary(double sal){annualSalary=sal;}

void Employee::setAnnualSalary(string sal){annualSalary=atof(sal.c\_str());}

int Employee::getNumEmployees(){return numEmployees;}

void Employee::setBenefits(Benefits bn){bens = bn;}

Benefits Employee::getBenefits(){return bens;}

// create Salaried Class

class Salaried : public Employee{

private:

const int MIN\_MANAGEMENT\_LEVEL = 0;

const int MAX\_MANAGEMENT\_LEVEL = 3;

const double BONUS\_PERCENT = 10;

int managementLevel;

public:

Salaried();

Salaried(string, string, char, int, double, Benefits, int);

Salaried(double, int);

double calculatePay(double);

void displayEmployee();

void setManagementLevel(int ml);

int getManagementLevel();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Salaried Class Functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Salaried::Salaried()

:

Employee(){

managementLevel = 0;

}

Salaried::Salaried(string fn, string ln, char gen, int dep, double sal, Benefits ben, int ml)

:

Employee(fn, ln, gen, dep, sal, ben)

{

managementLevel = ml;

}

Salaried::Salaried(double sal, int ml)

:

Employee()

{

annualSalary = sal;

managementLevel = ml;

}

double Salaried::calculatePay(double sal){

double bonus = managementLevel\*BONUS\_PERCENT;

sal= sal+bonus;

double weeklyPay= sal/52;

return weeklyPay;

}

void Salaried::displayEmployee(){

cout<<"\nEmployee Information"<<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl<<endl;

cout<<"Employee Name: \t"<<firstName<<" "<<lastName<<endl;

cout<<"Gender: \t"<<gender<<endl;

cout<<"Dependents: \t"<<dependents<<endl;

cout<<"Annual Salary: \t"<< setprecision(2) << showpoint << fixed << annualSalary<<endl;

cout<<"Weekly Salary: \t"<< setprecision(2) << showpoint << fixed <<calculatePay(annualSalary)<<endl;

bens.displayBenefits();

cout<<"Employee Type:\t Salaried"<<endl;

cout<<"Management Level: \t"<<managementLevel<<endl;

cout<<endl<<endl;

}

void Salaried::setManagementLevel(int ml){

while(ml < MIN\_MANAGEMENT\_LEVEL || ml > MAX\_MANAGEMENT\_LEVEL){

cout<<"Error: You must input a number greater than -1 or less than 4"<<endl;

}

managementLevel = ml;

}

int Salaried::getManagementLevel(){return managementLevel;}

// create Hourly Class

class Hourly : public Employee{

private:

const double MIN\_WAGE = 10;

const double MAX\_WAGE = 75;

const double MIN\_HOURS = 0;

const double MAX\_HOURS = 50;

double wage;

double hours;

string category;

public:

Hourly();

Hourly(double, double, string);

Hourly(string, string, char, int, double, double, Benefits, string);

double calculatePay(double sal);

void displayEmployee();

void setWage(double wg);

double getWage();

void setHours(double hrs);

double getHours();

void setCategory(string cat);

string getCategory();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Hourly Class Functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Hourly::Hourly():Employee(){

wage = 0.0;

hours = 0.0;

category = "X";

}

Hourly::Hourly(double wg, double hrs, string cat){

wage = wg;

hours = hrs;

category = cat;

}

Hourly::Hourly(string fn, string ln, char gen, int dep, double wg, double hrs, Benefits ben, string cat)

:Employee(fn, ln, gen, dep, ben)

{

wage= wg;

hours = hrs;

category = cat;

}

double Hourly::calculatePay(double sal){

// not sure what to do with this either!

}

void Hourly::displayEmployee(){

cout<<"\nEmployee Information"<<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"<<endl<<endl;

cout<<"Employee Name: \t"<<firstName<<" "<<lastName<<endl;

cout<<"Gender: \t"<<gender<<endl;

cout<<"Dependents: \t"<<dependents<<endl;

cout<<"Annual Salary: \t"<< setprecision(2) << showpoint << fixed << annualSalary<<endl;

cout<<"Weekly Salary: \t"<< setprecision(2) << showpoint << fixed <<calculatePay(annualSalary)<<endl;

bens.displayBenefits();

cout<<"Employee Type:\t Hourly"<<endl;

cout<<"Category: \t"<<category<<endl;

cout<<"Wage: \t"<<wage<<endl;

cout<<"Hours: \t"<<hours<<endl;

cout<<endl<<endl;

}

void Hourly::setWage(double wg){}

double Hourly::getWage(){return wage;}

void Hourly::setHours(double hrs){hours = hrs;}

double Hourly::getHours(){return hours;}

void Hourly::setCategory(string cat){category = cat;}

string Hourly::getCategory(){return category;}

int main(){

// call function for header information

DisplayApplicationInformation();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Start Program\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// call function for divider for start program

DisplayDivider("Start Program");

DisplayDivider("Employee 1");

Salaried sal;

sal.setFirstName(getInput("First Name: "));

sal.setLastName(getInput("Last Name: "));

sal.setGender(getInput("Gender: ")[0]);

sal.setDependents(getInput("Number of Dependents: "));

// I can't figure out how to call the benefits now that they are protected...

sal.setAnnualSalary(getInput("Annual Salary: "));

// create variable to change user input to correct data type

string input;

input = getInput("Management Level: ");// use the input variable to store the users string input

int ml = atoi(input.c\_str());// change the string input to an integer and place inside of the ml variable

sal.setManagementLevel(ml);// set instance inside the Salaried class to ml

// call class function to display employee information

sal.displayEmployee();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End Program\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// call function to leave program

TerminateApplication();

// keep page open until user hits any key

cin.ignore();

return 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Non Class Functions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// create header function

void DisplayApplicationInformation(){

// create header to introduce the program

cout<<"\n--------------------------------------------"<<endl;

cout<<"| Welcome the Basic User Interface Program |"<<endl;

cout<<"| CIS247, Week 5 Lab |"<<endl;

cout<<"| Name: RENEE THOMAS |"<<endl;

cout<<"| This program adds a Salaried and Hourly |"<<endl;

cout<<"| class, then outputs the data for each. |"<<endl;

}

// create divider string function

void DisplayDivider(string outputTitle){

cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* "<<outputTitle<< " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl;

}

// create function to get information from user

string getInput(string inputType){

// declare variable strInput

string strInput;

// get information from user

cout<<"Enter your " <<inputType;

//put user input into variable strInput

getline(cin, strInput);

//output user input

return strInput;

}

// create good bye message

void TerminateApplication(){

cout<<"\n\nThank you for running the Employee class program for hourly and salaried employees."<<endl;

cout<<"Press any key to leave the program..."<<endl;

}