CPSC301 - Assignment 3

Description: A program is needed for a third-party payroll clearinghouse that supports the following:

- A calculate employee pay feature
- A company total pay feature

The employee and company totals are stored in a text file for each individual company.

1. Create the following “input.txt” file. Each record consists of a first name, last name, employee ID, company name, hours worked, and pay rate per hour.

```
Maria    Brown      10 Intel      42.75 39.0
Jeffrey  Jackson    20 Boeing     38.0  32.5
Bernard  Smith      30 Douglas    25.0  23.25
Matthew  Davenport  40 Raytheon   63.15 47.5
Kimberly Macias     50 Douglas    45.5  40.0
Amber    Daniels    60 Raytheon   62.25 44.5
Michael  Lee        70 Boeing     32.0  35.5
Patricia Wright     80 Intel      40.0  32.0
Stanley  Johnson    90 HealthTech 48.5  43.25
James    Miller    100 Raytheon   55.5  45.75
Linda    Davis     110 HealthTech 40.0  36.5
Olivia   Walker    120 Intel      61.15 51.5
Thomas   Sanders   130 Raytheon   50.25 38.5
Sophia   Foster    140 Raytheon   44.15 44.0
Ryan     Ward      150 Douglas    68.0  48.25
Karen    Hill      160 Intel      49.5  54.0
```

2. Each record will be saved in an instance of a CLASS called **Person**. The person class will be declared as follows **(HINT: this should go in a file called person.h)**:

```
//begin person.h
#ifndef PERSON_H
#define PERSON_H

#include <string>
using namespace std;

class Person
{
private:
    string firstName;
```

```
    string lastName;
    int    employeeID;
    string companyName;
    float  hoursWorked;
    float  payRate;

public:
    Person();
    void   setFirstName(string fName);
    string getFirstName();
    void   setLastName(string lName);
    string getLastName();
    void   setEmployeeId(int id);
    int    getEmployeeId();
    void   setCompanyName(string coName);
    string getCompanyName();
    void   setPayRate(float rate);
    float  getPayRate();
    void   setHoursWorked(float hours);
    float  getHoursWorked();
    float  totalPay();
    string fullName();
};
#endif // end person.h
```

3. You will need to make a separate file called **person.cpp**. The file person.cpp contains proper definitions for all function declarations in the header file. The first line of the file is an include statement:

```
#include "person.h"
```

Here is an example of a function definition in person.cpp:

```
string Person::getFirstName() {
    return firstName;
}
```

Note the **<ret type> Classname::funcName()** structure here. The "**::**" is referred to as a scope resolution operator. It lets the compiler know that this code is defining a function for the class **Person**, and not a bare function that can be used anywhere. It also lets the compiler know that the variables reference in the function can be found within that class. This is required, because they are private otherwise.

4. Make a new file called **pay.cpp** that includes:
- A main() function
- The #include "person.cpp" statement
- Any other necessary includes and/or helper functions
- All the additional details mentioned below

5. A vector called **employees** used to store the employee data when reading from the file (employees will be of the **Person** class). Use of the vector class (instead of an array) will ensure that your data space can grow as needed to handle any reasonable number of employees. Use of a vector is **required**. It should be declared in **main()**.

6. A vector called **companyNames** that holds the string names of each company (**companyNames** will be of type string). This information will be needed to complete step 10. Use of a vector is **required**. It should be declared in **main()**.

7. A function **readData** to read the data from “input.txt” and store it in the **employees** vector. NOTE: the vector needs to be passed to the function by reference and the file should be read only once in the program.

8. A function **getCompanies** that retrieves the company names from **employees** and adds them to **companyNames**. NOTE: both vectors need to be passed to this function by reference.

9. A function called **printHighestPaid** must find and output (cout) the full name, employee ID, company name, and total pay of the person who was paid the highest amount this statement. Call the function from main before exiting the program. NOTE: the **employees** vector needs to be passed to this function by reference. Example output:

```
Highest paid: Ryan Ward
Employee ID: 150
Employer: Douglas
Total Pay: $3281.00
```

10. A function called **separateAndSave** must write the payroll information to multiple text files - one for each company. Each file should be named after the company (e.g. Boeing.txt) and contain only the employees from that company. Within the file, the data should be formatted by selecting reasonable column widths for output. The float output should be truncated to 2 decimal places. The column widths do not have to match exactly. Simply pick a reasonable number of characters for each (20-30 for names, etc). The output should also have the total pay for each company. NOTE: both vectors need to be passed to this function by reference.

Intel.txt

```
Maria    Brown   10 Intel $1667.25
Patricia Wright  80 Intel $1280.00
Olivia   Walker 120 Intel $3149.22
Karen    Hill   160 Intel $2673.00
Total $8769.47
```

Boeing.txt

```
Jeffrey Jackson 20 Boeing $1235.00
Michael Lee     70 Boeing $1136.00
Total $2371.00
```

Douglas.txt

```
Bernard  Smith   30 Douglas $581.25
Kimberly Macias  50 Douglas $1820.00
Ryan     Ward   150 Douglas $3281.00
Total $5682.25
```

Raytheon.txt

```
Matthew  Davenport  40 Raytheon $2999.62
Amber    Daniels    60 Raytheon $2770.12
James    Miller    100 Raytheon $2539.12
Thomas   Sanders   130 Raytheon $1934.62
Sophia   Foster    140 Raytheon $1942.60
Total $12186.08
```

HealthTech.txt

```
Stanley Johnson  90 HealthTech $2097.62
Linda   Davis   110 HealthTech $1460.00
Total $3557.62
```

Additional Notes:

- No global variables are allowed
- Do not change function/class/vector/file names or function declaration
- You may only call the open file method to read the file one time in the program

- Vectors must be used

Optional Enhancements:

- Step 8: consider an implementation which would improve the efficiency of the function
- Overload the insertion and extraction operators for class Person; update the **readData** and **separateAndSave** functions to use these operators
    - Implement a function that can use fstream and iostream interchangeably