Evan Nguyen
Joseph Guzman
CECS 275 - 03/25
Group #12

Lab #4

Main.cpp

```cpp
/*
 * Answer to Lab 4
 * CECS 275 - Spring 2022
 * @author Joseph Guzman
 * @author Evan Nguyen
 * @version 1.0.0
 *
 */

#include <iostream>
#include <fstream>
#include <vector>
#include <iomanip>
#include "Employee.h"
#include "BankAccount.h"

using namespace std;

int main()
{
    //Vector for each of the employees within the txt file
    vector<Employee*> employees;

    //Points for Employee and Bank Account
    Employee *employ;
    BankAccount *bank_account;

    //Initialize variables
    string id_Num;
    string first_Name;
    string last_Name;
    string word;
    string starting_Salary;
    string temp_Salary;
    double salary;

    // Read from the text file
    ifstream fileread("salary.txt");

    // Use a while loop together with the getline() function to read the file line by line
    while (fileread >> id_Num)
    {
        // reading first name, last name and salary into string type variables
        fileread >> first_Name;
        fileread >> last_Name;
        fileread >> starting_Salary;

        // converts string to double
        salary = stod(starting_Salary);

        // creates a bank account for the Employee
        bank_account = new BankAccount();

        // setting bank account id same as Employee id
        bank_account->set_id(id_Num);

```

```
55          bank_account->set_id(id_Num);
56
57          // creating a new Employee object
58          employ = new Employee(id_Num, first_Name, last_Name, salary, bank_account);
59
60          // push back into vector "employees"
61          employees.push_back(employ);
62      }
63      // closing file
64      fileread.close();
65
66      // Create and open a text file for writing
67      ofstream filewrite("monthly_salary.txt");
68
69      filewrite << "Employee ID   |Last Name       |First Name       |Annual Salary    |Monthly Salary    |Balance   " << endl;
70
71      // write the rest to a file
72
73      // Close the file
74      filewrite.close();
75      return 0;
76  }
77
78
```

Employee.h

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab4 > deliver > Employee.h > ...

```h
#ifndef EMPLOYEE_H
#define EMPLOYEE_H

#include "BankAccount.h"
#include <string>

// omit using namespace std in header files
// and implicitly type std

class Employee {

    public:
        /*
        Constructs an employee with a id, given name, salary, and
        bank account.
        @param id the id
        @param n the name
        @param s the annual salary
        @param a a pointer to the bank account
        */
        employee(std::string id, std::string fn, std::string ln, double s, BankAccount* a);

        //Deposits one month's salary into the bank account.
        void deposit_monthly_salary();

        //Prints this employee's information to cout.
        void print() const;

        // GET ID
        std::string get_id() const;

        // GET first name
        std::string get_fn() const;

        // GET last name
        std::string get_ln() const;

        // GET salary
        double get_salary() const;

        // GET - pointer to employees account instance
        BankAccount *get_account() const;

    private:
        std::string id;
        // separate name to fn and ln cuz its easier
        std::string thisFn;
        std::string thisLn;
        double thisSalary;
        BankAccount* account;
};

#endif
```

Employee.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab4 > deliver > Employee.cpp > ...

```cpp
#include "Employee.h"
#include <string>
#include <iostream>

// implementation file, so we can use this here
using namespace std;

// Constructor function - set every new instance of Employee (via reading file)
Employee::employee(string id, string fn, string ln, double s, BankAccount* a){
    // read id, name, salary from file
    this->id = id;
    thisFn = fn;
    thisLn = ln;
    thisSalary = s;
    account = a;
}

//Deposits one month's salary into the bank account.
void Employee::deposit_monthly_salary(){
    double newSal = thisSalary / 12;
    account->deposit(newSal);
}

/*
Prints this employee's information to cout.
Void - dont return and only set a member variable
*/
void Employee::print() const{
    cout << "ID: " << id << endl;
    cout << "FN: " << thisFn << endl;
    cout << "LN: " << thisLn << endl;
    cout << "Salary: " << thisSalary << endl;
    cout << "Bank ID: " << account->get_id() << endl;
    cout << "Account balance: " << account->get_balance() << endl;
}

string Employee::get_id() const {

    return id;
}

// GET first_name, set as constant to avoid changes
// Taken from file and initialized in instance creation
string Employee::get_fn() const{
    return thisFn;
}

// GET last_name, set as constant to avoid changes
// Taken from file and initialized in instance creation
string Employee::get_ln() const {
    return thisLn;
}

// GET salary, set as constant to avoid changes
// Taken from file and initialized in instance creation
double Employee::get_salary() const {
```

```cpp
double Employee::get_salary() const {
    return thisSalary;
}

// GET account, set as constant to avoid changes
// Point to account
BankAccount *Employee::get_account() const {
    return account;
}

```

Bankaccount.h

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab4 > de

```cpp
#ifndef BANK_ACCOUNT_H
#define BANK_ACCOUNT_H

#include <string>

// omit using namespace std in header files
// and implicitly type std

class BankAccount {
    public:
        //Constructs a bank account with a $2000 balance.
        BankAccount();

        /* DEPOSIT
        Deposits money into this account.
        @param amount the amount to deposit.
        */
        void deposit(double amount);

        /*
        Withdraws money from this account.
        @param amount the amount to withdraw.
        */
        void withdraw(double amount);

        /*
        Gets the balance of this account.
        @return the balance
        */
        double get_balance() const;

        /*
        Gets the id of this account.
        @return the balance
        */
        std::string get_id() const;

        /*
        Set id for bank account.
        @return the balance
        */
        void set_id(std::string id);

    private:
        std::string id;
        double balance;
};

#endif
```

Bankaccount.cpp

C: > Users > nguye > Documents > OneDrive > CLASS > CECS 275 > CECS-275-LABS > Lab4 > deliver > BankAccount.cpp >

```cpp
#include "BankAccount.h"
#include <string>

// implementation file, so we can use this here
using namespace std;

// Constructor Function - set every new instance of BankAccount (via employee)
BankAccount::BankAccount(){

    // get id from employee
    id;

    // default bank balance
    balance = 2000;

}

/* DEPOSIT
Deposits money into this account.
@param amount the amount to deposit.
Void - dont return and only set a member variable
*/
void BankAccount::deposit(double amount){
    balance = balance + amount;
}

/*
Withdraws money from this account.
@param amount the amount to withdraw.
*/
void BankAccount::withdraw(double amount){
    // not needed?
}

/*
Gets the balance of this account.
@return the balance
*/
double BankAccount::get_balance() const{
    // to set balance, point to it

    return balance;
}

/*
Gets the id of this account.
@return the id
*/
string BankAccount::get_id() const{

    return id;
}

/*
Set id for bank account.
Void - dont return and only set a member variable
```

We were unable to finish the lab and produce an output