HAS ERRORS

#include <iostream>

#include <fstream>

#include <string>

#include "Company.h"

using namespace std;

#define MAIN() void main()

class Company;

class FileHelper

{

public:

static string filename;

static void Save(string cv)

{

ofstream fout(filename, ios::app);

if (fout.is\_open())

{

fout << cv << endl;

fout.close();

}

}

static Company Read()

{

Company jp;

ifstream fin(filename);

if (fin.is\_open())

{

string companyName;

fin >> companyName;

jp.companyName = companyName;

int count = 0;

fin >> count;

jp.count = count;

for (int i = 0; i < count; i++)

{

Applier\* w = new Applier;

string name;

string surname;

string speciality;

string about;

fin >> name >> surname >> speciality >> about;

w->SetName(name);

w->SetSurname(surname);

w->SetSpeciality(speciality);

w->SetAbout(about);

jp.AddApplier(\*w);

}

jp.ShowAppliers();

fin.close();

return jp;

}

}

};

string FileHelper::filename = "cvs.txt";

MAIN()

{

Applier a("John", "Johnlu", "Programmer", "I have 2 years of experience");

string cv1 = a.GetCV();

cout << cv1 << endl;

Applier a2("Memmed", "Memmedli", "Designer", "I have 6 years of experience");

string cv2 = a2.GetCV();

Company c("ITSTEPMMC");

c.HireUp(cv1);

c.HireUp(cv2);

c.ShowCVs();

}

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

#define MAIN() void main()

#pragma region WriteReadToTextFile

//void dataWriteToFile(string filename, string text)

//{

// ofstream fout(filename, ios::out);

// if (fout.is\_open())

// {

// fout << text;

// }

// else

// {

// cout << filename << " does not exist!" << endl;

// }

// fout.close();

//}

//

//void dataWriteToFileApp(const string& filename, const string& text)

//{

// ofstream fout(filename, ios::app);

// if (fout.is\_open())

// {

// fout << text;

// }

// else

// {

// cout << filename << " does not exist!" << endl;

// }

// fout.close();

//}

//

//string GetDataFromFile(const string& filename)

//{

// ifstream fin(filename, ios::in);

// string result = "";

// if (fin.is\_open())

// {

// //getline(fin, result); one line

//

// string data = "";

// while (!fin.eof()) // eof - end of file

// {

// //fin >> data;

// getline(fin, data);

// result.append(data + "\n");

// }

// }

// else

// {

// throw string("This file " + filename + " does not exist!");

// }

// fin.close();

// return result;

//}

//int main()

//{

// string text = "Hello world";

// //dataWriteToFile("Mario.txt", text);

// //dataWriteToFileApp("Mario.txt", "\nMario is the best character");

// try

// {

// string result = GetDataFromFile("Mario.txt");

// cout << result << endl;

// }

// catch (const string& ex)

// {

// cout << "Error : " << ex << endl;

// }

//

// return 0;

//}

#pragma endregion

#pragma region WriteReadToBinaryFile

//

//void WriteDataToBinaryFile()

//{

// ofstream fout("obj.bin", ios\_base::binary | ios\_base::app);

//

// int obj[5]{ 1,2,3,4,5 };

// if (fout.is\_open())

// {

// fout.write((char\*)obj, sizeof(int) \* 5);

// }

// fout.close();

//}

//

//void ReadDataFromBinaryFile()

//{

// ifstream fin("obj.bin", ios\_base::binary);

// int arr[5]{};

// if (fin.is\_open())

// {

// fin.read((char\*)arr, sizeof(int) \* 5);

// }

// fin.close();

// for (int x = 0; x < 5; x++)

// {

// cout << arr[x] << " ";

// }

// cout << endl;

//

//}

//

//

//int main()

//{

// WriteDataToBinaryFile();

// ReadDataFromBinaryFile();

//

// return 0;

//}

#pragma endregion

class Worker

{

string name;

int age;

public:

Worker() {}

Worker(const string& name, const int& age)

{

SetName(name);

SetAge(age);

}

int GetAge() const

{

return age;

}

string GetName() const

{

return name;

}

void SetName(const string& name)

{

this->name = name;

}

void SetAge(const int& age)

{

this->age = age;

}

void Show() const

{

cout << "WORKER INFO" << endl;

cout << "Name : " << name << endl;

cout << "Age : " << age << endl;

}

};

class JobPosting

{

public:

string companyName;

Worker\*\* appliers;

int count = 0;

JobPosting() {}

JobPosting(const string& companyName)

{

this->companyName = companyName;

}

Worker\*\* GetAppliers() const

{

return appliers;

}

int GetApplierCount() const

{

return count;

}

void AddApplier(const Worker& applier)

{

Worker\*\* newappliers = new Worker \* [count + 1]{};

for (size\_t i = 0; i < count; i++)

{

newappliers[i] = appliers[i];

}

if (count != 0)

{

delete[]appliers;

}

newappliers[count] = new Worker(applier);

count++;

appliers = newappliers;

newappliers = nullptr;

}

void ShowAppliers() const

{

cout << "Company name : " << companyName << endl;

cout << "==== Appliers ==== " << endl;

for (int i = 0; i < count; i++)

{

appliers[i]->Show();

cout << "==================" << endl;

}

}

};

class FileHelper

{

public:

static string filename;

static void Save(JobPosting jp)

{

ofstream fout(filename, ios::app);

if (fout.is\_open())

{

fout << jp.companyName << endl;

fout << jp.count << endl;

for (int i = 0; i < jp.count; i++)

{

fout << jp.appliers[i]->GetName() << " " << jp.appliers[i]->GetAge() << endl;

}

fout.close();

}

}

static JobPosting Read()

{

JobPosting jp;

ifstream fin(filename);

if (fin.is\_open())

{

string companyName;

fin >> companyName;

jp.companyName = companyName;

int count = 0;

fin >> count;

jp.count = count;

for (int i = 0; i < count; i++)

{

Worker\* w = new Worker;

string name;

int age;

fin >> name >> age;

w->SetName(name);

w->SetAge(age);

jp.AddApplier(\*w);

}

jp.ShowAppliers();

}

}

};

string FileHelper::filename = "job.txt";

MAIN()

{

Worker w1("Gulya", 25);

Worker w2("Aysel", 23);

Worker w3("Aysen", 32);

Worker w4("Rafiq", 25);

JobPosting jp("ITSTEPMMC");

jp.AddApplier(w1);

jp.AddApplier(w2);

jp.AddApplier(w3);

jp.AddApplier(w4);

//jp.ShowAppliers();

FileHelper::Save(jp);

JobPosting obj;

obj = FileHelper::Read();

obj.ShowAppliers();

}