计算机与网络安全学院

软件工程系

项目名称 C++综合程序设计

题　　目 员工信息管理系统

学生学号 201641404313

学生姓名 吴俊鸿

指导教师 肖捷老师

完成时间 2017 年 06 月 01 日　至　2017 年 06 月 13 日

项目地址 https://github.com/TonyNgcn/WorkerManagement/

目　录

1、系统分析……

1.1 功能需求分析……

1.2 数据需求分析……

2、系统设计……

2.1 系统功能模块设计……

2.2 类的抽象与类间关系……

2.3 类的设计……

3、系统编码……

3.1 程序文件模块划分……

3.2 程序代码……

4、系统运行结果……

5、总结……

5.1 自我评价及收获……

5.2 有待解决的问题及进一步完善的思路……

1. 系统分析

1.1 功能需要分析

该软件用于管理某公司的经理、技术员、销售员、销售经理 4 类人员信息，人员信息包括：工号、姓名、性别、部门、岗位、出生日期、当月工资等，具体功能包括：

① 添加功能：添加部门和员工信息
② 查询功能：提供多种组合查询方式，查询各类人员信息和部门信息。
③ 删除功能：删除各类人员信息，以及删除部门信息。
④ 显示功能：显示输出所有人员信息。
⑤ 修改功能：修改人员信息和部门名称
⑥ 分析功能：分析各项员工工资数据

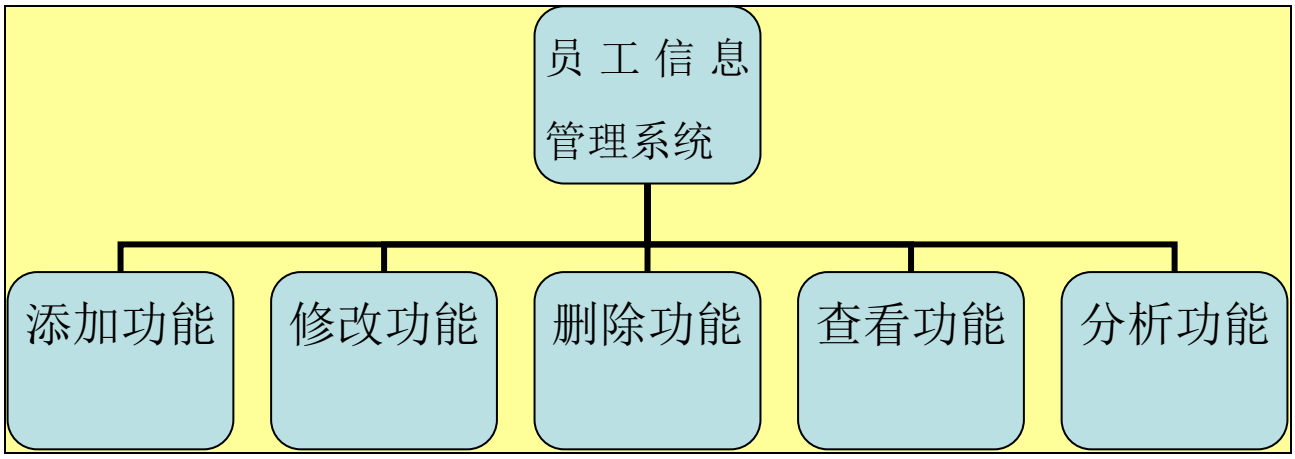
1.2 数据需要分析

该软件用于管理某公司的经理、技术员、销售员、销售经理 4 类人员信息，人员信息包括：工号、姓名、性别、部门、岗位、出生日期、当月工资等，各数据项含义如下：

- 工号：表示员工的编号。
- 姓名：表示员工的姓名。
- 性别：表示员工的性别。
- 出生日期：表示员工的出生日期，包括年、月、日 3 个数据分量，它是一个复杂数据项。
- 部门：表示员工的部门。（数字表示）
- 岗位：表示员工的工作岗位。//1-销售员 2-技术员 3-销售经理 4-经理
- 当月工资：表示员工的当月工资。
- 工作时间：表示技术员的工作时间。
- 销售额：表示销售员的销售额。

2. 系统设计

2.1 系统功能模块设计



2．类的抽象与类间关系

“员工信息管理”系统，抽象出几个类——department 类、date 类、basicInfo 类、interFace 类，其中：

- department 类，部门类，表示一个部门
- interFace 类,界面类，包含菜单及数据的处理
- date 类：日期类，表示一个日期
- basicInfo 类：表示人员类，包括经理、技术员、销售员、销售经理 4 类人员。

类间关系——组合关系，如下图如下。

2．3 类设计

“员工信息管理”系统的类设计。

2.3.1 部门类

```
class department
{
    string depName;    //部门名称
    int depNo;         //部门编号
    int count;         //部门人数
public:
    //函数功能：提取部门名称
    string getDepName()const;

    //函数功能：提取部门编号
    int getDepNo()const;

    //函数功能：输出部门名称、编号
    void printAllDep()const;

    //函数功能：输出该部门名称、编号、人数
    void getDepInfo()const;

    //函数功能：输入部门编号、名称
    void input();

    //函数功能：部门人数加一
    bool addCount();

    //函数功能：提取部门人数
    int getCount()const;

    //函数功能：构造函数
    department();
```

```
    //函数功能：修改部门名称
    bool changeDepName();

    //函数功能：部门人数减一
    bool reduceCount();

    //函数功能：检查部门名称是否重复
    bool checkDepName(string toCheck);

    //函数功能：检查部门编号是否重复
    bool checkDepNo(int toCheck);

    //函数功能：计算该部门平均工资
    double calAverageSalary();

    //函数功能：重载输入运算符
    friend istream& operator >>(istream&, department &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream&, department &);
};
```

2.3.2日期类

```
class date
{
    int year;    //年
    int month;   //月
    int day;     //日
public:
    //函数功能：构造函数
    date(int, int, int);

    //函数功能：已存在日期直接录入
    void setDate(int, int, int);

    //函数功能：重载输入运算符
    friend istream& operator >>(istream &, date &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream &, date &);

    //函数功能：输出日期数据
    void print()const;

    //函数功能：输入日期数据
    void input();
```

```
};
```

2.3.3 基本信息类

```
class basicInfo
{
protected:
    int no;         //员工工号
    string name;    //姓名
    string sex;     //性别
    int department;//部门编号
    date birthday; //出生日期
    double salary; //当月工资
    int workPost;  //1-销售员 2-技术员 3-销售经理 4-经理
public:
    //函数功能：构造函数
    basicInfo();

    //函数功能：提取工号
    int getNo()const;

    //函数功能：提取姓名
    string getName()const;

    //函数功能：提取性别
    string getSex()const;

    //函数功能：提取部门编号
    int getDepartment()const;

    //函数功能：提取工作岗位编号
    int getWorkPost()const;

    //函数功能：输出出生日期
    void getDate()const;

    //函数功能：提取出生日期
    date getBirthday()const;

    //函数功能：提取当月工资
    double getSalary()const;

    //函数功能：虚函数，输入数据
    virtual void input();

    //函数功能：设置部门编号
    void inputDepNo(int depNo);
```

```
//函数功能：纯虚函数，输出单个员工信息
virtual void printSingle() = 0;

//函数功能：纯虚函数，输出多个员工信息（无表头）
virtual void printNoHead() = 0;

//函数功能：纯虚函数，计算工资
virtual void calSalary() = 0;

//函数功能：重载输入运算符
friend istream& operator >>(istream&, basicInfo &);

//函数功能：重载输出运算符
friend ostream& operator <<(ostream&, basicInfo &);

//函数功能：重载相等运算符
bool operator ==(basicInfo &a)const;

//函数功能：比较工资多少
static bool bigger(const basicInfo *a,const basicInfo *b);

//函数功能：检查员工号是否重复
bool checkNo(int toCheck)const;

//函数功能：修改姓名
bool changeName();

//函数功能：纯虚函数，修改工作时间
virtual bool changeWorkTime() = 0;

//函数功能：纯虚函数，修改销售额
virtual bool changeSaleAmount() = 0;

//函数功能：修改性别
bool changeSex();

//函数功能：修改出生日期
bool changeBirthday();

//函数功能：修改部门编号
bool changeDep(int depToChange);

//函数功能：检查姓名是否合法
bool checkName(string toCheck)const;

//函数功能：设置已知信息
```

```
    bool setBasicInfo(int, string, string, int, date);
};
```

2.3.4 经理类　从基本信息类继承

```
class manager :public basicInfo
{
public:
    //函数功能：构造函数
    manager();

    //函数功能：输入
    void input();

    //函数功能：输出一个员工数据
    void printSingle();

    //函数功能：输出员工员工数据（无表头）
    void printNoHead();

    //函数功能：计算员工工资
    void calSalary();

    //函数功能：没有用
    bool changeSaleAmount();

    //函数功能：没有用
    bool changeWorkTime();

    //函数功能：重载输入运算符
    friend istream& operator >>(istream&, manager &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream&, manager &);
};
```

2.3.5 销售员类　从基本信息类继承

```
class salesman :public basicInfo
{
    double saleAmount;   //销售额
public:
    //函数功能：构造函数
    salesman();

    //函数功能：提取销售额
    double getSaleAmount();

    //函数功能：输入数据
```

```
    void input();

    //函数功能：输出一个员工数据
    void printSingle();

    //函数功能：输出一个员工数据（无表头）
    void printNoHead();

    //函数功能：计算工资
    void calSalary();

    //函数功能：更改销售额
    bool changeSaleAmount();

    //函数功能：没有用
    bool changeWorkTime();

    //函数功能：重载输入运算符
    friend istream& operator >>(istream&, salesman &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream&, salesman &);
};
```

2.3.6 销售经理类 从基本信息类继承

```
class salesmanager :public basicInfo
{
public:
    //函数功能：构造函数
    salesmanager();

    //函数功能：输入数据
    void input();

    //函数功能：输出一个员工信息
    void printSingle();

    //函数功能：输出一个员工信息（无表头）
    void printNoHead();

    //函数功能：计算工资
    void calSalary();

    //函数功能：没有用
    bool changeSaleAmount();

    //函数功能：没有用
```

```
    bool changeWorkTime();

    //函数功能：重载输入运算符
    friend istream& operator >>(istream&, salesmanager &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream&, salesmanager &);
};
```

2.3.7 技术员类　从基本信息类继承

```
class technician :public basicInfo
{
    int workHour;   //工作时间
public:
    //函数功能：构造函数
    technician();

    //函数功能：提取工作时间
    int getWorkHour();

    //函数功能：输入数据
    void input();

    //函数功能：输出一个员工信息
    void printSingle();

    //函数功能：输出一个员工信息（无表头）
    void printNoHead();

    //函数功能：计算工资
    void calSalary();

    //函数功能：修改工作时间
    bool changeWorkTime();

    //函数功能：没有用
    bool changeSaleAmount();

    //函数功能：重载输入运算符
    friend istream& operator >>(istream&, technician &);

    //函数功能：重载输出运算符
    friend ostream& operator <<(ostream&, technician &);
};
```

2.3.8 界面类

```
class interFace
{
```

```
public:
    //部门对象临时变量
    department tempDep;

    //员工基类指针临时变量
    basicInfo *tempPerson;

    //储存员工基类指针临时容器
    vector<basicInfo*> temp_v;

    //保存部门的容器
    vector<department> department_v;

    //保存经理的容器
    vector<manager> manager_v;

    //保存销售员的容器
    vector<salesman> salesman_v;

    //保存销售经理的容器
    vector<salesmanager> salesmanager_v;

    //保存技术员的容器
    vector<technician> technician_v;

    //函数功能：主菜单
    void mainMenu();

    //函数功能：用于增加的菜单
    void addMenu();

    //函数功能：用于修改的菜单
    void changeMenu();

    //函数功能：用于查看的菜单
    void checkMenu();

    //函数功能：用于删除的菜单
    void deleteMenu();

    //函数功能：用于分析的菜单
    void analysisMenu();

    //函数功能：修改部门名称
    void changeDepName();
```

```
//函数功能：修改员工信息
void changePersonInfo();

//函数功能：计算全部工资
void calAllSalary();

//函数功能：计算全体平均工资
double calAverageSalary();

//函数功能：删除部门
bool deleteDep(department);

//函数功能：输出所有部门
void getAllDep()const;

//函数功能：输出制定部门的名称
void getDepName(int)const;

//函数功能：增加部门
void addDep();

//函数功能：增加员工
void addPerson();

//函数功能：增加指定部门的员工数量
void addCountOfDep(int depNo);

//函数功能：判断该部门是否存在
bool checkDepExist(int checkDepID)const;

//函数功能：判断该部门是否存在销售经理
bool checkSalesManager(int checkDepID)const;

//函数功能：根据部门编号查找部门
bool searchDep(int checkDepID);

//函数功能：根据部门名称查找部门
bool searchDep(string checkDepName);

//函数功能：输出找到部门的信息
void checkByDep();

//函数功能：把四种不同岗位的员工放入一个基类指针的容器
void tempAll();

//函数功能：对基类指针的容器进行排序
```

```
    void sortAll();

    //函数功能：根据姓名查找员工
    bool searchByName(string checkName);

    //函数功能：根据工号查找员工
    bool searchByNo(int checkNo);

    //函数功能：更改工作岗位
    void changeWorkPost();

    //函数功能：减少该部门的人数
    void reduceDepCount(int depNo);

    //函数功能：删除员工
    bool deletePerson();

    //函数功能：把数据从容器读入文件
    bool vectorToFile();

    //函数功能：把数据从文件读入容器
    bool fileToVector();

    //函数功能：分页显示员工信息
    void printByPages();

    //函数功能：计算销售员平均工资
    double calSalesmanAverageSalary();

    //函数功能：计算销售经理平均工资
    double calSalesmanagerAverageSalary();

    //函数功能：计算技术员平均工资
    double calTechnicianAverageSalary();
```

};3. 系统编码

3．1 程序文件模块划分

员工信息管理”系统的程序文件模块的划分。

- All.h：全部类定义文件，定义全部类的结构。
- main.cpp：主控模块，程序入口，用于调用 interface 类的相关功能。
- interface.cpp：界面类实现文件，定义 interFace 类的成员函数的具体实现。
- department.cpp：部门类实现文件，定义 department 类的成员函数的具体实现。
- person.cpp：基本信息类实现文件，定义 date、 basicInfo 类的成员函数的具体实现。
- salesman.cpp：销售员类实现文件，定义 salesman 类的成员函数的具体实现。

- technician.cpp：技术员类实现文件，定义 technician 类的成员函数的具体实现。
- salesmanager. cpp：销售经理类实现文件，定义 salesmanager 类的成员函数的具体实现。
- manager.cpp：经理类实现文件，定义 manager 类的成员函数的具体实现。

3．2 程序代码

“员工信息管理”系统的程序代码。

3.2.1 All.h：全部类定义文件，定义全部类的结构。

All.h 与 2.3 中的代码一致。

3.2.2 main.cpp：主控模块，程序入口。

```
#include "ALL.h"
interFace inter;
int main()
{
    inter.fileToVector();
    while (1)
        inter.mainMenu();
    return 0;

}
```

3.2.3 interface.cpp：界面类实现文件

```
#include "ALL.h"

void interFace::mainMenu()
{
    system("cls");
    cout << "员工信息管理系统" << endl
        << "==============================" << endl;
    cout << "1.增加部门信息或员工信息" << endl
        << "2.修改部门信息或员工信息" << endl
        << "3.查询部门信息或员工信息" << endl
        << "4.删除部门信息或员工信息" << endl
        << "5.统计分析部门或员工信息" << endl
        << "6.保存信息并退出系统" << endl;
    cout << "请输入对应序号（1-6）：";
    int choice = 0;
    cin >> choice;
    cin.clear();
    cin.ignore(100, '\n');
    while (choice < 1 || choice>6)
    {
        cout << "输入错误，请重新选择：";
        cin >> choice;
```

```
            cin.clear();
            cin.ignore(100, '\n');
        }
        if (choice == 1)
            addMenu();
        else if (choice == 2)
            changeMenu();
        else if (choice == 3)
            checkMenu();
        else if (choice == 4)
            deleteMenu();
        else if (choice == 5)
            analysisMenu();
        else
        {
            system("cls");
            cout << "员工信息管理系统——保存信息并退出系统" << endl
                << "========================================" << endl;
            if (!vectorToFile())
            {
                cout << "文件保存不成功" << endl;
                system("pause");
            }
            exit(0);
        }
}

void interFace::addMenu()
{
    while (1)
    {
        system("cls");
        cout << "员工信息管理系统——增加部门信息或员工信息" << endl
            << "========================================" << endl;
        cout << "注意：添加员工前需要先添加部门信息" << endl
            << "1.添加部门信息" << endl
            << "2.添加员工信息" << endl
            << "3.返回上一层菜单" << endl;
        cout << "请输入对应序号（1-3）：";
        int choice = 0;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        while (choice < 1 || choice>3)
        {
            cout << "输入错误，请重新选择：";
```

```
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (choice == 1)
            addDep();
        else if (choice == 2)
            addPerson();
        else
            return;
    }
}

void interFace::changeMenu()
{
    while (1)
    {
        system("cls");
        cout << "员工信息管理系统——修改部门信息或员工信息" << endl
            << "========================================" << endl;
        cout << "注意：员工工号和部门编号均不可修改" << endl
            << "1.修改部门名字" << endl
            << "2.修改员工信息" << endl
            << "3.返回上一层菜单" << endl;
        cout << "请输入对应序号（1-3）：";
        int choice = 0;
        int menuChoice = 0;
        cin >> menuChoice;
        cin.clear();
        cin.ignore(100, '\n');
        while (menuChoice < 1 || menuChoice>3)
        {
            cout << "输入错误，请重新选择：";
            cin >> menuChoice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (menuChoice == 1)
            changeDepName();
        else if (menuChoice == 2)
            changePersonInfo();
        else if (menuChoice == 3)
            return;
    }
```

```
}

void interFace::checkMenu()
{
	while (1)
	{
		system("cls");
		cout << "员工信息管理系统——查询部门信息或员工信息" << endl
			<< "=========================================" << endl;
		cout << "1.查看已有部门" << endl
			<< "2.根据部门编号查询部门员工信息" << endl
			<< "3.根据部门名字查询部门员工信息" << endl
			<< "4.分页查询所有员工信息（按工资高低输出）" << endl
			<< "5.查询所有销售员信息" << endl
			<< "6.查询所有技术员信息" << endl
			<< "7.查询所有销售经理信息" << endl
			<< "8.查询所有经理信息" << endl
			<< "9.返回上一层菜单" << endl;
		cout << "请输入对应序号（1-9）：";
		int choice = 0;
		int menuChoice = 0;
		cin >> menuChoice;
		cin.clear();
		cin.ignore(100, '\n');
		while (menuChoice < 1 || menuChoice>9)
		{
			cout << "输入错误，请重新选择：";
			cin >> menuChoice;
			cin.clear();
			cin.ignore(100, '\n');
		}
		system("cls");
		if (menuChoice == 1)
		{
			for (auto &i : department_v)
				i.getDepInfo();
		}
		else if (menuChoice == 2)
		{
			cout << "请输入要查询的部门编号,输入 0 返回：";
			choice = -1;
			cin >> choice;
			cin.clear();
			cin.ignore(100, '\n');
			if (!choice)
				return;
```

```
        while (choice < 0 || !searchDep(choice))
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        tempDep.getDepInfo();
        tempAll();
        cout << "工号   姓名   性别   部门    出生日期" << endl;
        for (auto &i : temp_v)
        {
            if (i->getDepartment() == tempDep.getDepNo())
            {
                i->printNoHead();
            }
        }
    }
    else if (menuChoice == 3)
    {
        string depNameToCheck;
        cin >> depNameToCheck;
        if (!searchDep(depNameToCheck))
        {
            cout << "找不到该部门，请核对" << endl;
        }
        else
        {
            system("cls");
            tempDep.getDepInfo();
            tempAll();
            cout << "工号   姓名   性别   部门    出生日期" << endl;
            for (auto &i : temp_v)
            {
                if (i->getDepartment() == tempDep.getDepNo())
                {
                    i->printNoHead();
                }
            }
        }
    }
    else if (menuChoice == 4)
    {
        printByPages();
    }
```

```
        else if (menuChoice == 5)
        {
            cout << "工号   姓名   性别   部门    出生日期" << endl;
            for (auto &i : salesman_v)
                i.printNoHead();
            cout << "=========================" << endl;
            cout << "全部信息已显示完全" << endl;
        }
        else if (menuChoice == 6)
        {
            cout << "工号   姓名   性别   部门    出生日期" << endl;
            for (auto &i : technician_v)
                i.printNoHead();
            cout << "=========================" << endl;
            cout << "全部信息已显示完全" << endl;
        }
        else if (menuChoice == 7)
        {
            cout << "工号   姓名   性别   部门    出生日期" << endl;
            for (auto &i : salesmanager_v)
                i.printNoHead();
            cout << "=========================" << endl;
            cout << "全部信息已显示完全" << endl;
        }
        else if (menuChoice == 8)
        {
            cout << "工号   姓名   性别   部门    出生日期" << endl;
            for (auto &i : manager_v)
                i.printNoHead();
            cout << "=========================" << endl;
            cout << "全部信息已显示完全" << endl;
        }
        else if (menuChoice == 9)
        {
            return;
        }
        system("pause");
    }
}

void interFace::deleteMenu()
{
    while (1)
    {
        system("cls");
        cout << "员工信息管理系统——删除部门信息或员工信息" << endl
```

```
        << "=========================================" << endl;
    cout << "注意：删除部门会同步删除该部门的员工信息" << endl;
    cout << "1.删除部门信息" << endl
        << "2.删除员工信息" << endl
        << "3.返回上一层" << endl;
    cout << "请输入对应序号（1-3）：";
    int choice = 0;
    int menuChoice = 0;
    cin >> menuChoice;
    cin.clear();
    cin.ignore(100, '\n');
    while (menuChoice < 1 || menuChoice>3)
    {
        cout << "输入错误，请重新选择：";
        cin >> menuChoice;
        cin.clear();
        cin.ignore(100, '\n');
    }
    system("cls");
    if (menuChoice == 1)
    {
        cout << "请输入要删除的部门编号,输入 0 返回：";
        choice = -1;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        if (!choice)
            return;
        while (choice < 0 || !searchDep(choice))
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        tempDep.getDepInfo();
        tempAll();
        cout << "工号　姓名　性别　部门　　出生日期" << endl;
        for (auto &i : temp_v)
        {
            if (i->getDepartment() == tempDep.getDepNo())
            {
                i->printNoHead();
            }
        }
```

```
        cout << "=======================================" << endl;
        cout << "确定删除该部门及所有员工信息，请按 1 确定：";
        choice = 0;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        if (choice == 1)
            if (deleteDep(tempDep))
                cout << "删除成功" << endl;
            else
                cout << "删除失败" << endl;
    }
    else if (menuChoice == 2)
    {
        cout << "目前可以通过员工工号或姓名查找你要修改的员工" << endl
            << "1-姓名  2-工号" << endl;
        cout << "请选择对应编号，输入 0 可以返回上一层：";
        choice = -1;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        while (choice < 0 || choice>2)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (!choice)
            continue;
        else if (choice == 1)
        {
            cout << "请输入员工姓名：";
            string name;
            cin >> name;
            if (!searchByName(name))
            {
                cout << "姓名输入错误，程序会返回";
                system("pause");
                continue;
            }
        }
        else if (choice == 2)
        {
            cout << "请输入员工工号：";
```

```
                int no = 0;
                cin >> no;
                if (!searchByNo(no) || !no)
                {
                    cout << "工号输入错误，程序将返回";
                    system("pause");
                    continue;
                }
            }
            system("cls");
            tempPerson->printSingle();
            cout << "确定删除该员工的信息，请按 1 确定：";
            choice = 0;
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
            if (choice == 1)
                if (deletePerson())
                    cout << "删除成功" << endl;
                else
                    cout << "删除失败" << endl;
        }
        else if(menuChoice ==3)
            return;
    }
}

void interFace::analysisMenu()
{
    while(1)
    {
        system("cls");
        cout << "员工信息管理系统——查询部门信息或员工信息" << endl
            << "========================================" << endl;
        cout << "1.统计并显示某个部门的平均月工资、最低月工资、最高月工资" << endl
            << "2.统计并显示某个部门超出本部门平均月工资的人数与员工信息" << endl
            << "3.统计并显示所有员工中的最低月工资和最高月工资员工的信息" << endl
            << "4.统计并显示所有员工超出平均月工资的人数与员工信息" << endl
            << "5.查看各岗位的平均工资" << endl
            << "6.返回上一层菜单" << endl;
        cout << "请输入对应序号（1-6）：";
        int choice = 0;
        int menuChoice = 0;
        cin >> menuChoice;
        cin.clear();
        cin.ignore(100, '\n');
```

```
while (menuChoice < 1 || menuChoice>6)
{
	cout << "输入错误，请重新选择：";
	cin >> menuChoice;
	cin.clear();
	cin.ignore(100, '\n');
}
system("cls");
if (menuChoice == 1)
{
	cout << "请输入要查询的部门编号,输入 0 返回：";
	choice = -1;
	cin >> choice;
	cin.clear();
	cin.ignore(100, '\n');
	double maxSalary = 0;
	double minSalary = 0;
	if (!choice)
		continue;
	while (choice < 0 || !searchDep(choice))
	{
		cout << "输入错误，请重新选择：";
		cin >> choice;
		cin.clear();
		cin.ignore(100, '\n');
	}
	system("cls");
	if (tempDep.getCount() == 0)
	{
		cout << "该部门为空" << endl;
		system("pause");
		continue;
	}
	tempDep.getDepInfo();
	tempAll();
	calAllSalary();
	for (auto &i : temp_v)
	{
		if (i->getDepartment() == tempDep.getDepNo())
		{
			maxSalary = i->getSalary();
			minSalary = i->getSalary();
			break;
		}
	}
	for (auto &i : temp_v)
```

```
        {
            if (i->getDepartment() == tempDep.getDepNo())
            {
                if(maxSalary < i->getSalary())
                    maxSalary = i->getSalary();
                if (minSalary > i->getSalary())
                    minSalary = i->getSalary();
            }
        }
        cout << "平均工资：" << tempDep.calAverageSalary() << endl
            << "最低工资：" << minSalary << endl
            << "最高工资：" << maxSalary << endl;
    }
    if (menuChoice == 2)
    {
        cout << "请输入要查询的部门编号,输入 0 返回：";
        choice = -1;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        if (!choice)
            continue;
        int count = 0;
        while (choice < 0 || !searchDep(choice))
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (tempDep.getCount() == 0)
        {
            cout << "该部门为空" << endl;
            system("pause");
            continue;
        }
        tempDep.getDepInfo();
        tempAll();
        calAllSalary();
        double averageSalary = tempDep.calAverageSalary();
        cout << "工号　姓名　性别　部门　　出生日期" << endl;
        for (auto &i : temp_v)
        {
            if (i->getDepartment() == tempDep.getDepNo())
            {
```

```
                if (i->getSalary() >= averageSalary)
                {
                    i->printNoHead();
                    count++;
                }
            }
        }
        cout << "=========================================" << endl;
        cout << "全部信息已显示完全,超过平均工资的人数是" << count << endl;
    }
    else if (menuChoice == 3)
    {
        double maxSalary = 0;
        double minSalary = 0;
        tempAll();
        calAllSalary();
        for (auto &i : temp_v)
        {
            maxSalary = i->getSalary();
            minSalary = i->getSalary();
            break;
        }
        for (auto &i : temp_v)
        {
            if (maxSalary < i->getSalary())
                maxSalary = i->getSalary();
            if (minSalary > i->getSalary())
                minSalary = i->getSalary();
        }
        cout << "平均工资：" << setprecision(2) << fixed << calAverageSalary() << endl
            << "最低工资：" << setprecision(2) << fixed << minSalary << endl
            << "最高工资：" << setprecision(2) << fixed << maxSalary << endl;
    }
    else if (menuChoice == 4)
    {
        int count = 0;
        tempAll();
        calAllSalary();
        double averageSalary = tempDep.calAverageSalary();
         cout << "工号  姓名  性别  部门    出生日期" << endl;
        for (auto &i : temp_v)
        {
            if (i->getSalary() >= averageSalary)
            {
                count++;
                i->printNoHead();
```

```
                }
            }
            cout << "=========================================" << endl;
            cout << "全部信息已显示完全,超过平均工资的人数是" << count << endl;
        }
        else if (menuChoice == 5)
        {
            cout << "销售员平均工资：" << setprecision(2) << fixed << calSalesmanAverageSalary() << endl
                << "技术员平均工资：" << setprecision(2) << fixed << calTechnicianAverageSalary() << endl
                << "销售经理平均工资：" << setprecision(2) << fixed << calSalesmanagerAverageSalary() << endl
                << "经理平均工资：8000" << endl;
        }
        else if(menuChoice ==6)
            return;
        system("pause");
    }
}

void interFace::changeDepName()
{
    while(1)
    {
        system("cls");
        getAllDep();
        cout << "请输入要修改的部门编号,输入 0 返回：";
        int choice = -1;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        if (!choice)
            return;
        while (choice < 0 || !searchDep(choice))
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        for(auto &i:department_v)
        {
            if (i.getDepNo() == tempDep.getDepNo())
            {
                i.changeDepName();
```

```
                cout << "修改成功" << endl;
                break;
            }
        }
    }

}

void interFace::changePersonInfo()
{
    cout << "目前可以通过员工工号或姓名查找你要修改的员工" << endl
        << "1-姓名  2-工号" << endl;
    cout << "请选择对应编号，输入 0 可以返回上一层：";
    int choice = -1;
    cin >> choice;
    cin.clear();
    cin.ignore(100, '\n');
    while (choice < 0 || choice>2)
    {
        cout << "输入错误，请重新选择：";
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
    }
    system("cls");
    if (!choice)
        return;
    else if (choice == 1)
    {
        cout << "请输入员工姓名：";
        string name;
        cin >> name;
        if (!searchByName(name))
        {
            cout << "姓名输入错误，程序会返回" << endl;
            system("pause");
            return;
        }
    }
    else
    {
        cout << "请输入员工工号：";
        int no = 0;
        cin >> no;
        if(!searchByNo(no)||!no)
        {
```

```
            cout << "工号输入错误，程序将返回" << endl;
            system("pause");
            return;
        }
    }
    while (1)
    {
        system("cls");
        tempPerson->printSingle();
        cout << "选择要修改的项目" << endl;
        cout << "1-姓名  2-性别  3-出生日期  4-部门  5-岗位  ";
        if (tempPerson->getWorkPost() == 1)
            cout << "6-销售额" << endl;
        else if (tempPerson->getWorkPost() == 2)
            cout << "6-工作时间" << endl;
        else
            cout << endl;
        cout << "请选择（输入 0 返回）：";
        choice = -1;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        while (choice < 0 || choice>6)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        if (!choice)
            return;
        if (choice == 1)
            tempPerson->changeName();
        else if (choice == 2)
            tempPerson->changeSex();
        else if (choice == 3)
            tempPerson->changeBirthday();
        else if (choice == 4)
        {
            getAllDep();
            bool exist = false;
            cout << "请输入部门编号：";
            choice = -1;
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
```

```
                while (choice < 1 || !searchDep(choice))
                {
                    cout << "输入错误，请重新选择：";
                    cin >> choice;
                    cin.clear();
                    cin.ignore(100, '\n');
                }
                system("cls");
                if (!choice)
                    return;
                for (auto &i : department_v)
                {
                    if (i.getDepNo() == tempDep.getDepNo())
                    {
                        exist = true;
                        break;
                    }
                }
                if (!exist)
                {
                    cout << "该部门不存在，程序将返回" << endl;
                    system("pause");
                    return;
                }
                else
                    tempPerson->changeDep(choice);
            }
            else if (choice == 5)
                changeWorkPost();
            else if (choice == 6)
            {
                if (tempPerson->getWorkPost() == 1)
                    tempPerson->changeSaleAmount();
                else if (tempPerson->getWorkPost() == 2)
                    tempPerson->changeWorkTime();
                else
                {
                    cout << "输入错误，程序将返回" << endl;
                    system("pause");
                    return;
                }
            }
        }
}

void interFace::calAllSalary()
```

```
{
	tempAll();
	for (auto &i : temp_v)
	{
		i->calSalary();
	}
}

double interFace::calAverageSalary()
{
	calAllSalary();
	double averageSalary = 0;
	for (auto &i : temp_v)
	{
		averageSalary += i->getSalary();
	}
	if (!averageSalary)
		return 0;
	averageSalary /= temp_v.size();
	return averageSalary;
}

bool interFace::deleteDep(department delDep)
{
	for (vector<department>::iterator it=department_v.begin();it!=department_v.end();it++)
	{
		if ((*it).getDepNo() == delDep.getDepNo())
		{
			it = department_v.erase(it);
			for (vector<salesman>::iterator it = salesman_v.begin(); it != salesman_v.end(); )
			{
				if ((*it).getDepartment() == delDep.getDepNo())
				{
					it = salesman_v.erase(it);
				}
				else
					it++;
			}
			for (vector<salesmanager>::iterator it = salesmanager_v.begin(); it != salesmanager_v.end(); )
			{
				if ((*it).getDepartment() == delDep.getDepNo())
				{
					it = salesmanager_v.erase(it);
				}
				else
					it++;
```

```
                }
                for (vector<technician>::iterator it = technician_v.begin(); it != technician_v.end(); )
                {
                    if ((*it).getDepartment() == delDep.getDepNo())
                    {
                        it = technician_v.erase(it);
                    }
                    else
                        it++;
                }
                for (vector<manager>::iterator it = manager_v.begin(); it != manager_v.end(); )
                {
                    if ((*it).getDepartment() == delDep.getDepNo())
                    {
                        it = manager_v.erase(it);
                    }
                    else
                        it++;
                }
                return true;
            }
        }
        return false;
}

void interFace::getAllDep()const
{
        for (auto &i : department_v)
            i.printAllDep();
}

void interFace::getDepName(int depNo) const
{
        for (auto &i : department_v)
        {
            if (i.getDepNo() == depNo)
            {
                cout << i.getDepNo() << '-' << i.getDepName();
                break;
            }
        }
}

void interFace::addDep()
{
        department newDep;
```

```
    newDep.input();
    department_v.push_back(newDep);
}

void interFace::addPerson()
{
    cout << "添加员工" << endl;
    cout << "请选择员工所属部门" << endl;
    getAllDep();
    cout << "请输入部门编号：";
    int choice = 0;
    cin >> choice;
    cin.clear();
    cin.ignore(100, '\n');
    while (!choice)
    {
        cout << "输入错误，请重新选择：";
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
    }
    if (!checkDepExist(choice))
    {
        cout << "该部门不存在，请先添加部门，程序将返回" << endl;
        system("pause");
        return;
    }
    int tempDepNo=choice;
    choice = 0;
    if (checkSalesManager(tempDepNo))
    {
        cout << "请选择员工的职位：\n1.销售员\n2.技术员\n3.经理\n4.销售经理" << endl;
        cout << "请输入选择（1-4）：";
        cin >> choice;
        cin.clear();
        cin.ignore(100,'\n');
        while (choice < 1 || choice>4)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100,'\n');
        }
    }
    else
    {
```

```
        cout << "该部门已存在销售经理，所以员工的职位不可以是销售经理" << endl;
        cout << "请选择员工的职位：\n1.销售员\n2.技术员\n3.经理" << endl;
        cout << "请输入选择（1-3）：";
        cin >> choice;
        cin.clear();
        cin.ignore(100,'\n');
        while (choice < 1 || choice>3)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100,'\n');
        }
    }
    if (choice == 1)
    {
        salesman obj1;
        obj1.input();
        obj1.inputDepNo(tempDepNo);
        salesman_v.push_back(obj1);
    }
    else if (choice == 2)
    {
        technician obj2;
        obj2.input();
        obj2.inputDepNo(tempDepNo);
        technician_v.push_back(obj2);
    }
    else if (choice == 3)
    {
        manager obj3;
        obj3.input();
        obj3.inputDepNo(tempDepNo);
        manager_v.push_back(obj3);
    }
    else if (choice == 4)
    {
        salesmanager obj4;
        obj4.input();
        obj4.inputDepNo(tempDepNo);
        salesmanager_v.push_back(obj4);
    }
    else
    {
        cout << "输入错误，程序将返回" << endl;
        system("pause");
```

```
            return;
        }
        addCountOfDep(tempDepNo);
        system("pause");
        system("cls");
}

void interFace::addCountOfDep(int depNo)
{
        for (auto &i : department_v)
        {
            if (i.getDepNo() == depNo)
                i.addCount();
        }
}

bool interFace::checkDepExist(int checkDepID) const
{
        for (auto &i : department_v)
        {
            if (i.getDepNo() == checkDepID)
                return true;
        }
        return false;
}

bool interFace::checkSalesManager(int checkDepID) const
{
        for (auto &i : salesmanager_v)
        {
            if (i.getDepartment() == checkDepID)
                return false;
        }
        return true;
}

bool interFace::searchDep(int checkDepID)
{
        for (auto &i : department_v)
        {
            if (i.getDepNo() == checkDepID)
            {
                tempDep = i;
                return true;
            }
        }
```

```
    return false;
}

bool interFace::searchDep(string checkDepName)
{
    for (auto &i : department_v)
    {
        if (i.getDepName() == checkDepName)
        {
            tempDep = i;
            return true;
        }
    }
    return false;
}

void interFace::checkByDep()
{
    cout << "部门名称：" << tempDep.getDepNo() << '-' << tempDep.getDepName() << endl;
    cout << "部门人数：" << tempDep.getCount() << endl<<endl;

    for (auto &i : salesman_v)
    {
        if(i.getDepartment()==tempDep.getDepNo())
            i.printNoHead();
    }
    for (auto &i : technician_v)
    {
        if (i.getDepartment() == tempDep.getDepNo())
            i.printNoHead();
    }
    for (auto &i : salesmanager_v)
    {
        if (i.getDepartment() == tempDep.getDepNo())
            i.printNoHead();
    }
    for (auto &i : manager_v)
    {
        if (i.getDepartment() == tempDep.getDepNo())
            i.printNoHead();
    }

}

void interFace::tempAll()
{
```

```
    temp_v.swap(vector<basicInfo*>());
    for (auto &i : technician_v)
    {
        temp_v.push_back(&i);
    }
    for (auto &i : salesman_v)
    {
        temp_v.push_back(&i);
    }
    for (auto &i : salesmanager_v)
    {
        temp_v.push_back(&i);
    }
    for (auto &i : manager_v)
    {
        temp_v.push_back(&i);
    }
}

void interFace::sortAll()
{
    tempAll();
    sort(temp_v.begin(), temp_v.end(), basicInfo::bigger);
}

bool interFace::searchByName(string checkName)
{
    bool found = false;
    tempAll();
    for (auto &i : temp_v)
    {
        if (i->getName() == checkName)
        {
            tempPerson = i;
            found = true;
        }
    }
    temp_v.swap(vector<basicInfo*>());
    return found;
}

bool interFace::searchByNo(int checkNo)
{
    bool found = false;
    tempAll();
    for (auto &i : temp_v)
```

```
    {
        if (i->getNo() == checkNo)
        {
            tempPerson = i;
            found = true;
        }
    }
    temp_v.swap(vector<basicInfo*>());
    return found;
}

void interFace::changeWorkPost()
{
    cout << "请输入要修改到的岗位编号（1-销售员 2-技术员 3-销售经理 4-经理，0 返回）：";
    int choose = -1;
    cin >> choose;
    cin.clear();
    cin.ignore(100,'\n');
    while (choose < 1 && choose>4)
    {
        if (!choose)
        {
            return;
        }
        cout << "输入错误，请重新输入" << endl;
        cout << "岗位编号（输入 0 返回）：";
        cin >> choose;
        cin.clear();
        cin.ignore(100,'\n');
    }
    if (choose == tempPerson->getWorkPost())
        return;
    if (choose == 1)
    {
        salesman obj1;
        obj1.setBasicInfo(tempPerson->getNo(), tempPerson->getName(), tempPerson->getSex(),
tempPerson->getDepartment(), tempPerson->getBirthday());
        deletePerson();
        salesman_v.push_back(obj1);
    }
    if (choose == 2)
    {
        technician obj2;
        obj2.setBasicInfo(tempPerson->getNo(), tempPerson->getName(), tempPerson->getSex(),
tempPerson->getDepartment(), tempPerson->getBirthday());
        deletePerson();
```

```
            technician_v.push_back(obj2);
        }
        if (choose == 3)
        {
            if (!checkSalesManager(tempPerson->getDepartment()))
            {
                cout << "该部门已有销售经理，程序将不作修改，直接返回";
                system("pause");
                return;
            }
            salesmanager obj3;
            obj3.setBasicInfo(tempPerson->getNo(), tempPerson->getName(), tempPerson->getSex(),
tempPerson->getDepartment(), tempPerson->getBirthday());
            deletePerson();
            salesmanager_v.push_back(obj3);
        }
        if (choose == 4)
        {
            manager obj4;
            obj4.setBasicInfo(tempPerson->getNo(), tempPerson->getName(), tempPerson->getSex(),
tempPerson->getDepartment(), tempPerson->getBirthday());
            deletePerson();
            manager_v.push_back(obj4);
        }
}

void interFace::reduceDepCount(int depNo)
{
        for (auto &i : department_v)
        {
            if (i.getDepNo() == depNo)
                i.reduceCount();
        }
}

bool interFace::deletePerson()
{
        int depPerson = tempPerson->getWorkPost();
        for (auto &i : department_v)
        {
            if (tempPerson->getDepartment() == i.getDepNo())
            {
                i.reduceCount();
                break;
            }
        }
```

```
    if (depPerson == 1)
    {
        for (vector<salesman>::iterator it = salesman_v.begin(); it != salesman_v.end(); it++)
        {
            if (it->getNo() == tempPerson->getNo())
            {
                it = salesman_v.erase(it);
                return true;
            }
        }
    }
    else if (depPerson == 2)
    {
        for (vector<technician>::iterator it = technician_v.begin(); it != technician_v.end(); it++)
        {
            if (it->getNo() == tempPerson->getNo())
            {
                it = technician_v.erase(it);
                return true;
            }
        }
    }
    else if (depPerson == 3)
    {
        for (vector<salesmanager>::iterator it = salesmanager_v.begin(); it != salesmanager_v.end(); it++)
        {
            if (it->getNo() == tempPerson->getNo())
            {
                it = salesmanager_v.erase(it);
                return true;
            }
        }
    }
    else if (depPerson == 4)
    {
        for (vector<manager>::iterator it = manager_v.begin(); it != manager_v.end(); it++)
        {
            if (it->getNo() == tempPerson->getNo())
            {
                it = manager_v.erase(it);
                return true;
            }
        }
    }
    return false;
}
```

```
bool interFace::vectorToFile()
{
	system("cls");
	cout << "数据正在写入文件中，请稍后……" << endl;
	int countSalesman = 0;
	int countSalesmanager = 0;
	int countTechnician = 0;
	int countManager = 0;
	int countDep = 0;
	ofstream salesman_f("salesman.dat");
	for (auto &i : salesman_v)
	{
		salesman_f << i << endl;
		countSalesman++;
	}
	salesman_f.close();
	ofstream technician_f("technician.dat");
	for (auto &i : technician_v)
	{
		technician_f << i << endl;
		countTechnician++;
	}
	technician_f.close();
	ofstream salesmanager_f("salesmanager.dat");
	for (auto &i : salesmanager_v)
	{
		salesmanager_f << i << endl;
		countSalesmanager++;
	}
	salesmanager_f.close();
	ofstream manager_f("manager.dat");
	for (auto &i : manager_v)
	{
		manager_f << i << endl;
		countManager++;
	}
	manager_f.close();
	ofstream department_f("department.dat");
	for (auto &i : department_v)
	{
		department_f << i << endl;
		countDep++;
	}
	department_f.close();
	cout << "数据写入成功，共有" << countDep << "个部门，" << countSalesman << "个销售员，" <<
```

```
countTechnician << "个技术员，" << countSalesmanager << "个销售经理，" << countManager << "个经理。" << endl;
	system("pause");
	return true;
}

bool interFace::fileToVector()
{
	system("cls");
	cout << "数据正在从文件读取中，请稍后……" << endl;
	int countSalesman = 0;
	int countSalesmanager = 0;
	int countTechnician = 0;
	int countManager = 0;
	int countDep = 0;
	ifstream department_f("department.dat");
	if (!department_f)
		NULL;
	else
	{
		while (!department_f.eof())
		{
			department dep;
			department_f >> dep;
			countDep++;
			department_v.push_back(dep);
			if (department_f.eof())
			{
				countDep--;
				department_v.pop_back();
				break;
			}
		}
		department_f.close();
	}


	ifstream salesman_f("salesman.dat");
	if (!salesman_f)
		NULL;
	else
	{
		while (!salesman_f.eof())
		{
			salesman person;
			salesman_f >> person;
```

```
            countSalesman++;
            salesman_v.push_back(person);
            if (salesman_f.eof())
            {
                countSalesman--;
                salesman_v.pop_back();
                break;
            }
        }
        salesman_f.close();
}


ifstream technician_f("technician.dat");
if (!technician_f)
        NULL;
else
{
        while (!technician_f.eof())
        {
            technician person;
            technician_f >> person;
            countTechnician++;
            technician_v.push_back(person);
            if (technician_f.eof())
            {
                countTechnician--;
                technician_v.pop_back();
                break;
            }
        }
        technician_f.close();
}


ifstream salesmanager_f("salesmanager.dat");
if (!salesmanager_f)
        NULL;
else
{
        while (!salesmanager_f.eof())
        {
            salesmanager person;
            salesmanager_f >> person;
            countSalesmanager++;
            salesmanager_v.push_back(person);
```

```
            if (salesmanager_f.eof())
            {
                countSalesmanager--;
                salesmanager_v.pop_back();
                break;
            }
        }
        salesmanager_f.close();
    }


    ifstream manager_f("manager.dat");
    if (!manager_f)
        NULL;
    else
    {
        while (!manager_f.eof())
        {
            manager person;
            manager_f >> person;
            countManager++;
            manager_v.push_back(person);
            if (manager_f.eof())
            {
                countManager--;
                manager_v.pop_back();
                break;
            }
        }
        manager_f.close();
    }

    cout << "数据读取成功，共有" << countDep << "个部门，" << countSalesman << "个销售员，" << countTechnician << "个技术员，" << countSalesmanager << "个销售经理，" << countManager << "个经理。" << endl;
    system("pause");
    system("cls");
    return true;
}

void interFace::printByPages()
{
    sortAll();
    if (!temp_v.size())
        {
        cout << "无数据，请先添加数据" << endl;
```

```
            system("pause");
            return;
            }
        cout << "共读取" << temp_v.size() << "条记录。" << endl;
        cout << "记录会按照工资从高到低排序，请输入你希望一页打印几条记录" << endl;
        cout << "输入一个大于 0，小于或等于" << temp_v.size() << "的整数：";
        int numToPrint=0;
        int choice = 0;
        cin >> numToPrint;
        cin.clear();
        cin.ignore(100, '\n');
        while (!numToPrint || numToPrint > temp_v.size())
        {
            cout << "输入错误，请重新输入：";
            cin >> numToPrint;
            cin.clear();
            cin.ignore(100, '\n');
        }
        double totalPage = temp_v.size() / numToPrint;
        bool finished = false;
FirstPage:
        int numToCount = 0;
        int page = 1;
        int endOfCount = numToPrint - 1;
        cout << "工号　姓名　性别　部门　　出生日期" << endl;
        for (;numToCount!=temp_v.size(); numToCount++)
        {
            temp_v[numToCount]->printNoHead();
            if (numToCount == endOfCount)
            {
                numToCount++;
                finished = true;
                break;
            }
        }
        if (!finished || numToPrint == temp_v.size())
        {
            if (page == 1)
            {
                cout << endl << "全部信息已显示完全" << endl;
                temp_v.swap(vector<basicInfo*>());
                return;
            }
            else
            {
EndPage:
```

```
                cout << endl << "该页已显示完全" << endl;
                cout << "1-上一页      2-返回" << endl;
                cout << "请选择（1-2）：";
                int choice = 0;
                cin >> choice;
                cin.clear();
                cin.ignore(100, '\n');
                while (choice < 1 || choice>2)
                {
                    cout << "输入错误，请重新选择：";
                    cin >> choice;
                    cin.clear();
                    cin.ignore(100, '\n');
                }
                system("cls");
                if (choice == 1)
                    goto FrontPage;
                else
                {
                    temp_v.swap(vector<basicInfo*>());
                    return;
                }
            }
        }
        cout << endl << "该页已显示完全" << endl;
        cout << "1-下一页      2-返回" << endl;
        cout << "请选择（1-2）：";
        choice = 0;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        while (choice < 1 || choice>2)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (choice == 1)
        {
NextPage:
            page++;
            endOfCount += numToPrint;
NormalPage:
            finished = false;
```

```
        cout << "工号   姓名   性别   部门    出生日期" << endl;
        for (; numToCount != temp_v.size(); numToCount++)
        {
            temp_v[numToCount]->printNoHead();
            if (numToCount == endOfCount)
            {
                finished = true;
                numToCount++;
                break;
            }
        }
        if (!finished || page >= totalPage)
            goto EndPage;
        cout << endl << "该页已显示完全" << endl;
        cout << "1-上一页    2-下一页    3-返回" << endl;
        cout << "请选择（1-3）：";
        choice = 0;
        cin >> choice;
        cin.clear();
        cin.ignore(100, '\n');
        while (choice < 1 || choice>3)
        {
            cout << "输入错误，请重新选择：";
            cin >> choice;
            cin.clear();
            cin.ignore(100, '\n');
        }
        system("cls");
        if (choice == 1)
        {
FrontPage:
            page--;
            endOfCount -= numToPrint;
            if (page == 1)
                goto FirstPage;
            else
            {
                numToCount -= numToPrint;
                goto NormalPage;
            }
        }
        else if (choice == 2)
            goto NextPage;
        else
        {
            temp_v.swap(vector<basicInfo*>());
```

```
                return;
            }
        }
        else
        {
            temp_v.swap(vector<basicInfo*>());
            return;
        }
}

double interFace::calSalesmanAverageSalary()
{
    double salesmanAverageSalary=0;
    for (auto &i : salesman_v)
    {
        i.calSalary();
        salesmanAverageSalary += i.getSalary();
    }
    if (!salesmanAverageSalary)
        return 0;
    salesmanAverageSalary /= salesman_v.size();
    return salesmanAverageSalary;
}

double interFace::calSalesmanagerAverageSalary()
{
    double salesmanagerAverageSalary=0;
    for (auto &i : salesmanager_v)
    {
        i.calSalary();
        salesmanagerAverageSalary += i.getSalary();
    }
    if (!salesmanagerAverageSalary)
        return 0;
    salesmanagerAverageSalary /= salesmanager_v.size();
    return salesmanagerAverageSalary;
}

double interFace::calTechnicianAverageSalary()
{
    double technicianAverageSalary=0;
    for (auto &i : technician_v)
    {
        i.calSalary();
        technicianAverageSalary += i.getSalary();
    }
```

```
    if (!technicianAverageSalary)
        return 0;
    technicianAverageSalary /= technician_v.size();
    return    technicianAverageSalary;
}
```

3.2.4 department.cpp：部门类实现文件

```
#include "ALL.h"
extern interFace inter;
string department::getDepName() const
{
    return depName;
}

int department::getDepNo() const
{
    return depNo;
}

void department::printAllDep() const
{
    cout << depNo << '-' << depName << endl;
}

void department::getDepInfo() const
{
    cout << depNo << '-' << depName << " 人数：" << count << endl;
}

void department::input()
{
    cout << "部门编号：";
    int noToInput = 0;
    cin >> noToInput;
    cin.clear();
    cin.ignore(100,'\n');
    while (!noToInput || !checkDepNo(noToInput))
    {
        cout << "部门编号输入错误或与已有编号重复，请重新输入" << endl;
        cout << "部门编号：";
        cin >> noToInput;
        cin.clear();
        cin.ignore(100,'\n');
    }
```

```
    depNo = noToInput;
    cout << "部门名称：";
    cin >> depName;
    while(!checkDepName(depName))
    {
        cout << "部门名称：";
        cin >> depName;
    }
    system("pause");
    system("cls");
}

bool department::addCount()
{
    if (++count)
        return true;
    return false;
}

int department::getCount() const
{
    return count;
}

department::department()
{
    depName = "No Name";
    depNo = 0;
    count = 0;
}

bool department::changeDepName()
{
    cout << "部门名称：";
    string depNameToChange;
    cin >> depNameToChange;
    while (!checkDepName(depNameToChange))
    {
        cout << "部门名称：";
        cin >> depNameToChange;
    }
    depName = depNameToChange;
    return true;
}

bool department::reduceCount()
```

```
{
    if (--count)
        return true;
    return false;
}

bool department::checkDepName(string toCheck)
{
    if (toCheck.size() < 2 || toCheck.size() > 10)
    {
        cout << "名字过长或过短，请重新输入" << endl;
        return false;
    }
    for (auto &i : inter.department_v)
    {
        if (i.getDepName() == toCheck)
        {
            cout << "已有同名部门，请重新输入" << endl;
            return false;
        }

    }
    return true;
}

bool department::checkDepNo(int toCheck)
{
    for (auto &i : inter.department_v)
    {
        if (toCheck == i.getDepNo())
            return false;
    }
    return true;
}

double department::calAverageSalary()
{
    double averageSalary = 0;
    int count = 0;
    inter.calAllSalary();
    for (auto i : inter.temp_v)
    {
        if (i->getDepartment() == depNo)
        {
            count++;
            averageSalary += i->getSalary();
```

```
        }
    }
    averageSalary /= count;
    return averageSalary;
}

istream & operator>>(istream &in, department &a)
{
    int noToInput;
    string nameToInput;
    int countToInput;
    in >> noToInput >> nameToInput >> countToInput;
    a.depNo = noToInput;
    a.depName = nameToInput;
    a.count = countToInput;
    return in;
}

ostream & operator<<(ostream &out, department &a)
{
    out << a.depNo<<' ' << a.depName<<' ' << a.count;
    return out;
}
```

3.2.5 person.cpp：基本信息类实现文件

```
#include "ALL.h"
extern interFace inter;
date::date(int y=2017, int m=6, int d=1)
{
    year = y;
    month = m;
    day = 1;
}

void date::setDate(int y, int m, int d)
{
    year = y;
    month = m;
    day = d;
}

istream& operator>>(istream &in,date &d)
{
    int y, m, da;
```

```
    in >> y >> m >> da;
    d.year = y;
    d.month = m;
    d.day = da;
    return in;
}

ostream& operator<<(ostream &out,date &d)
{
    out << d.year << ' ' << d.month << ' ' << d.day;
    return out;
}

void date::print()const
{
    cout << year << '-' << month << '-' << day;
}

void date::input()
{
    year = 0, month = 0, day = 0;
    cout << "年：";
    cin >> year;
    cin.clear();
    cin.ignore(100,'\n');
    while (year < 1917 || year>2017)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "年：";
        cin >> year;
        cin.clear();
        cin.ignore(100,'\n');
    }
    cout << "月：";
    cin >> month;
    cin.clear();
    cin.ignore(100,'\n');
    while (month < 1 || month>12)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "月：";
        cin >> month;
        cin.clear();
        cin.ignore(100,'\n');
    }
    cout << "日：";
```

```
    cin >> day;
    cin.clear();
    cin.ignore(100,'\n');
    while (day < 1 || day>31)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "日：";
        cin >> day;
        cin.clear();
        cin.ignore(100,'\n');
    }
}

basicInfo::basicInfo()
{
    salary = 0;
    no = 0;
}

int basicInfo::getNo()const
{
    return no;
}

string basicInfo::getName()const
{
    return name;
}

string basicInfo::getSex()const
{
    return sex;
}

int basicInfo::getDepartment()const
{
    return department;
}

int basicInfo::getWorkPost() const
{
    return workPost;
}

void basicInfo::getDate()const
{
```

```
    birthday.print();
}

date basicInfo::getBirthday() const
{
    return birthday;
}

double basicInfo::getSalary()const
{
    return salary;
}

void basicInfo::input()
{
    cout << "员工工号：";
    int noToInput=0;
    int sexToChoose = 0;
    cin >> noToInput;
    cin.clear();
    cin.ignore(100,'\n');
    while (!noToInput || !checkNo(noToInput))
    {
        cout << "输入错误或与已有工号重复，请重新输入" << endl;
        cout << "员工工号：";
        cin >> noToInput;
        cin.clear();
        cin.ignore(100,'\n');
    }
    no = noToInput;
    cout << "员工姓名：";
    cin >> name;
    while (!checkName(name))
    {
        cout << "姓名：";
        cin >> name;
    }
    cout << "员工性别（1-男，2-女）请输入编号：";
    cin >> sexToChoose;
    cin.clear();
    cin.ignore(100,'\n');
    while (sexToChoose < 1 || sexToChoose > 2)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "员工性别（1-男，2-女）请输入编号：";
        cin >> sexToChoose;
```

```
            cin.clear();
            cin.ignore(100,'\n');
        }
        if (sexToChoose == 1)
            sex = "男";
        else
            sex = "女";
        cout << "出生日期：" << endl;
        birthday.input();
}

void basicInfo::inputDepNo(int depNo)
{
        department = depNo;
}

istream& operator>>(istream &in, basicInfo &a)
{
        int noToInput;
        string nameToInput;
        string sexToInput;
        int depToInput;
        int yearToInput;
        int monthToInput;
        int dayToInput;
        double salaryToInput;
        int workPostToInput;
        in >> noToInput >> nameToInput >> sexToInput >> depToInput >> yearToInput >> monthToInput >>
dayToInput >> salaryToInput >> workPostToInput;
        a.birthday.setDate(yearToInput, monthToInput, dayToInput);
        a.no = noToInput;
        a.name = nameToInput;
        a.sex = sexToInput;
        a.department = depToInput;
        a.salary = salaryToInput;
        a.workPost = workPostToInput;
        return in;
}

ostream& operator<<(ostream &out, basicInfo &a)
{
        out << a.no << ' ' << a.name << ' ' << a.sex << ' ' << a.department << ' ' << a.birthday << ' ' <<
a.salary << ' ' << a.workPost;
        return out;
}
```

```
bool basicInfo::checkNo(int toCheck) const
{
    for (auto &i : inter.salesman_v)
    {
        if (i.getNo() == toCheck)
            return false;
    }
    for (auto &i : inter.technician_v)
    {
        if (i.getNo() == toCheck)
            return false;
    }
    for (auto &i : inter.salesmanager_v)
    {
        if (i.getNo() == toCheck)
            return false;
    }
    for (auto &i : inter.manager_v)
    {
        if (i.getNo() == toCheck)
            return false;
    }
    return true;
}

bool basicInfo::changeName()
{
    cout << "请输入修改后的姓名：";
    string nameToChange;
    cin >> nameToChange;
    while (!checkName(nameToChange))
    {
        cout << "修改后的姓名：";
        cin >> nameToChange;
    }
    name = nameToChange;
    return true;
}

bool basicInfo::changeSex()
{
    if (sex == "男")
        sex = "女";
    else
        sex = "男";
    return true;
```

```
}

bool basicInfo::changeBirthday()
{
    cout << "新的出生日期：" << endl;
    birthday.input();
    return true;
}

bool basicInfo::changeDep(int depToChange)
{
    if (workPost == 3)
    {
        cout << "销售经理不支持直接修改部门，请先转换成其他岗位" << endl;
        system("pause");
        return false;
    }
    for (auto &i : inter.department_v)
    {
        if (i.getDepNo() == department)
            i.reduceCount();
    }
    department = depToChange;
    for (auto &i : inter.department_v)
    {
        if (i.getDepNo() == department)
                i.addCount();
    }
    return true;
}


bool basicInfo::checkName(string toCheck)const
{
    if (toCheck.size() < 2 || toCheck.size() > 10)
    {
        cout << "名字过长或过短，请重新输入" << endl;
        return false;
    }
    return true;
}

bool basicInfo::setBasicInfo(int noSet, string nameSet, string sexSet , int departmentSet, date
birthdaySet)
{
    no = noSet;
```

```
    name = nameSet;
    department = departmentSet;
    sex = sexSet;
    birthday = birthdaySet;
    return true;
}

bool basicInfo::operator==(basicInfo & a)const
{
    return no==a.no&&name==a.name;
}

bool basicInfo::bigger(const basicInfo * a, const basicInfo * b)
{
    return a->salary > b->salary;
}
```

3.2.6 salesman.cpp：销售员类实现文件

```
#include "ALL.h"
extern interFace inter;
salesman::salesman()
{
    workPost = 1;
    salary = 0;
    saleAmount = -1;
}

double salesman::getSaleAmount()
{
    return saleAmount;
}

void salesman::input()
{
    basicInfo::input();
    cout << "销售额：";
    saleAmount = -1;
    cin >> saleAmount;
    cin.clear();
    cin.ignore(100,'\n');
    while (saleAmount < 0)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "销售额：";
```

```
        cin >> saleAmount;
        cin.clear();
        cin.ignore(100,'\n');
    }
}

void salesman::printSingle()
{
    calSalary();
    cout << "员工号：" << no << endl
        << "姓名：" << name << endl
        << "性别：" << sex << endl
        << "部门：";
    inter.getDepName(department);
    cout << endl << "出生日期:";
    birthday.print();
    cout << endl;
    cout << "职务：销售员" << endl
        << "销售额：" << setprecision(2) << fixed << saleAmount << endl
        << "工资：" << setprecision(2) << fixed << salary << endl << endl;
}

void salesman::printNoHead()
{
    calSalary();
    cout << setw(3) << no << ' ' << setw(7) << name << ' ' << setw(3) << sex << ' ';
    inter.getDepName(department);
    cout << ' ';
    birthday.print();
    //cout << endl;
    cout << " 职务：销售员" << ' '
        << "销售额：" << setprecision(2) << fixed << saleAmount << ' '
        << "工资：" << setprecision(2) << fixed << salary << endl;
}

void salesman::calSalary()
{
    basicInfo::salary = saleAmount*0.04;
}

bool salesman::changeSaleAmount()
{
    cout << "销售额：";
    double saleAmountToChange = -1;
    cin >> saleAmountToChange;
    cin.clear();
```

```
    cin.ignore(100, '\n');
    while (saleAmountToChange<0)
    {
        cout << "输入错误，请重新输入销售额：";
        cin >> saleAmountToChange;
        cin.clear();
        cin.ignore(100, '\n');
    }
    saleAmount = saleAmountToChange;
    return true;
}

bool salesman::changeWorkTime()
{
    return false;
}

istream & operator>>(istream &in, salesman &a)
{
    int noToInput;
    string nameToInput;
    string sexToInput;
    int depToInput;
    int yearToInput;
    int monthToInput;
    int dayToInput;
    double salaryToInput;
    int workPostToInput;
    double salesAmountToInput;
    in >> noToInput >> nameToInput >> sexToInput >> depToInput >> yearToInput >> monthToInput >>
dayToInput >> salaryToInput >> workPostToInput >> salesAmountToInput;
    a.birthday.setDate(yearToInput, monthToInput, dayToInput);
    a.no = noToInput;
    a.name = nameToInput;
    a.sex = sexToInput;
    a.department = depToInput;
    a.salary = salaryToInput;
    a.workPost = workPostToInput;
    a.saleAmount = salesAmountToInput;
    return in;
}

ostream & operator<<(ostream &out, salesman &a)
{
    out << a.no << ' ' << a.name << ' ' << a.sex << ' ' << a.department << ' ' << a.birthday << ' ' <<
a.salary << ' ' << a.workPost << ' ' << a.saleAmount;
```

```
    return out;
}
```

3.2.7technician.cpp：技术员类实现文件

```
#include "ALL.h"
extern interFace inter;
technician::technician()
{
    workHour = -1;
    salary = 0;
    workPost = 2;
}

int technician::getWorkHour()
{
    return workHour;
}

void technician::input()
{
    basicInfo::input();
    workHour = -1;
    cout << "工作时间：";
    cin >> workHour;
    cin.clear();
    cin.ignore(100,'\n');
    while (workHour < 0)
    {
        cout << "输入错误，请重新输入" << endl;
        cout << "工作时间：";
        cin >> workHour;
        cin.clear();
        cin.ignore(100,'\n');
    }
}

void technician::printSingle()
{
    calSalary();
    cout << "员工号：" << no << endl
        << "姓名：" << name << endl
        << "性别：" << sex << endl
        << "部门：";
    inter.getDepName(department);
```

```
    cout << endl << "出生日期:";
    birthday.print();
    cout << endl;
    cout << "职务：技术员" << endl
        << "工作时间（小时）：" << workHour << endl
        << "工资：" << setprecision(2) << fixed << salary << endl << endl;
}

void technician::printNoHead()
{
    calSalary();
    cout << setw(3) << no << ' ' << setw(7) << name << ' ' << setw(3) << sex << ' ';
    inter.getDepName(department);
    cout << ' ';
    birthday.print();
    //cout << endl;
    cout << " 职务：技术员" << ' '
        << "工作时间（小时）：" << workHour << ' '
        << "工资：" << setprecision(2) << fixed << salary << endl;
}

void technician::calSalary()
{
    basicInfo::salary = workHour * 100;
}

bool technician::changeWorkTime()
{
    cout << "工作时间：";
    int workHourToChange = -1;
    cin >> workHourToChange;
    cin.clear();
    cin.ignore(100, '\n');
    while (workHourToChange<0)
    {
        cout << "输入错误，请重新输入工作时间：";
        cin >> workHourToChange;
        cin.clear();
        cin.ignore(100, '\n');
    }
    workHour = workHourToChange;
    return true;
}

bool technician::changeSaleAmount()
{
```

```
    return false;
}

istream & operator>>(istream &in, technician &a)
{
    int noToInput;
    string nameToInput;
    string sexToInput;
    int depToInput;
    int yearToInput;
    int monthToInput;
    int dayToInput;
    double salaryToInput;
    int workPostToInput;
    int workHourToInput;
    in >> noToInput >> nameToInput >> sexToInput >> depToInput >> yearToInput >> monthToInput >>
dayToInput >> salaryToInput >> workPostToInput >> workHourToInput;
    a.birthday.setDate(yearToInput, monthToInput, dayToInput);
    a.no = noToInput;
    a.name = nameToInput;
    a.sex = sexToInput;
    a.department = depToInput;
    a.salary = salaryToInput;
    a.workPost = workPostToInput;
    a.workHour = workHourToInput;
    return in;
}

ostream & operator<<(ostream &out, technician &a)
{
    out << a.no << ' ' << a.name << ' ' << a.sex << ' ' << a.department << ' ' << a.birthday << ' ' <<
a.salary << ' ' << a.workPost << ' ' << a.workHour;
    return out;
}
```

3.2.8 salesmanager.cpp：销售经理类实现文件

```
#include "ALL.h"
extern interFace inter;
void salesmanager::input()
{
    basicInfo::input();
}

void salesmanager::printSingle()
```

```
{
    calSalary();
    cout << "员工号：" << no << endl
        << "姓名：" << name << endl
        << "性别：" << sex << endl
        << "部门：";
    inter.getDepName(department);
    cout << endl << "出生日期:";
    birthday.print();
    cout << endl;
    cout << "职务：销售经理" << endl
        << "工资：" << setprecision(2) << fixed << salary << endl << endl;
}

void salesmanager::printNoHead()
{
    calSalary();
    cout << setw(3) << no << ' ' << setw(7) << name << ' ' << setw(3) << sex << ' ';
    inter.getDepName(department);
    cout << ' ';
    birthday.print();
    //cout << endl;
    cout << " 职务：销售经理" << ' '
        << "工资：" << setprecision(2) << fixed << salary << endl;
}

void salesmanager::calSalary()
{
    double totalAmount = 0;
    for (auto &i : inter.salesman_v)
    {
        if (i.getDepartment() == department)
        {
            totalAmount += i.getSaleAmount();
        }
    }
    salary = 5000 + totalAmount*0.005;
}

bool salesmanager::changeSaleAmount()
{
    return false;
}

bool salesmanager::changeWorkTime()
{
```

```
    return false;
}

istream & operator>>(istream &in, salesmanager &a)
{
    int noToInput;
    string nameToInput;
    string sexToInput;
    int depToInput;
    int yearToInput;
    int monthToInput;
    int dayToInput;
    double salaryToInput;
    int workPostToInput;
    in >> noToInput >> nameToInput >> sexToInput >> depToInput >> yearToInput >> monthToInput >>
dayToInput >> salaryToInput >> workPostToInput;
    a.birthday.setDate(yearToInput, monthToInput, dayToInput);
    a.no = noToInput;
    a.name = nameToInput;
    a.sex = sexToInput;
    a.department = depToInput;
    a.salary = salaryToInput;
    a.workPost = workPostToInput;
    return in;
}

ostream & operator<<(ostream &out, salesmanager &a)
{
    out << a.no << ' ' << a.name << ' ' << a.sex << ' ' << a.department << ' ' << a.birthday << ' ' <<
a.salary << ' ' << a.workPost;
    return out;
}

salesmanager::salesmanager()
{
    workPost = 3;
    salary = 0;

}
```

3.2.9manager.cpp：经理类实现文件

```
#include "ALL.h"
extern interFace inter;
manager::manager()
{
    salary = 8000;
```

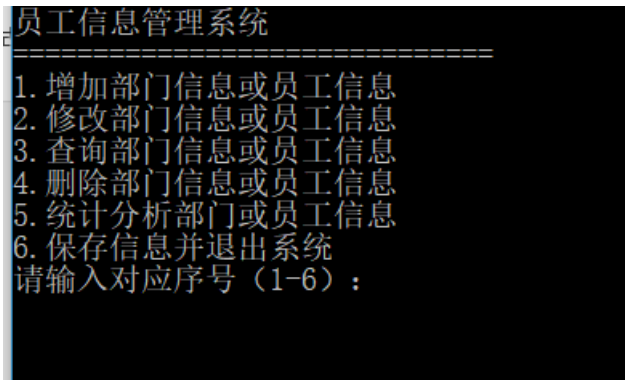
```
    workPost = 4;
}

void manager::input()
{
    basicInfo::input();
}

void manager::printSingle()
{
    calSalary();
    cout << "员工号：" << no << endl
        << "姓名：" << name << endl
        << "性别：" << sex << endl
        << "部门：";
    inter.getDepName(department);
    cout << endl << "出生日期:";
    birthday.print();
    cout << endl;
    cout << "职务：经理" << endl
        << "工资：" << setprecision(2) << fixed << salary << endl << endl;
}

void manager::printNoHead()
{
    calSalary();
    cout << setw(3) << no << ' ' << setw(7) << name << ' ' << setw(3) << sex << ' ';
    inter.getDepName(department);
    cout << ' ';
    birthday.print();
    //cout << endl;
    cout << " 职务：经理  " << ' '
        << "工资：" << setprecision(2) << fixed << salary << endl;
}

void manager::calSalary()
{
    basicInfo::salary = 8000;
}

bool manager::changeSaleAmount()
{
    return false;
}

bool manager::changeWorkTime()
```

```
{
    return false;
}

istream & operator>>(istream &in, manager &a)
{
    int noToInput;
    string nameToInput;
    string sexToInput;
    int depToInput;
    int yearToInput;
    int monthToInput;
    int dayToInput;
    double salaryToInput;
    int workPostToInput;
    in >> noToInput >> nameToInput >> sexToInput >> depToInput >> yearToInput >> monthToInput >>
dayToInput >> salaryToInput >> workPostToInput;
    a.birthday.setDate(yearToInput, monthToInput, dayToInput);
    a.no = noToInput;
    a.name = nameToInput;
    a.sex = sexToInput;
    a.department = depToInput;
    a.salary = salaryToInput;
    a.workPost = workPostToInput;
    return in;
}

ostream & operator<<(ostream &out, manager &a)
{
    out << a.no << ' ' << a.name << ' ' << a.sex << ' ' << a.department << ' ' << a.birthday << ' ' <<
a.salary << ' ' << a.workPost;
    return out;
}
```

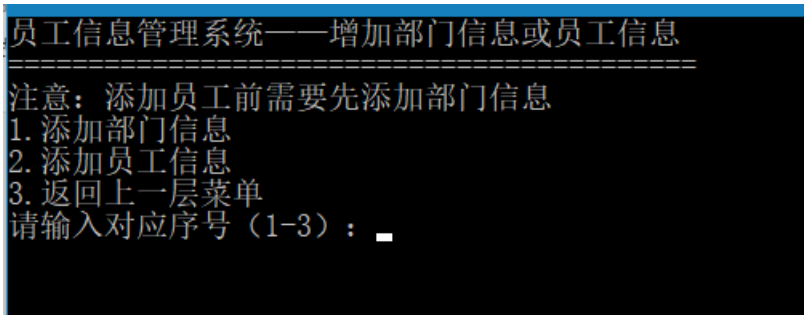
4. 系统运行结果

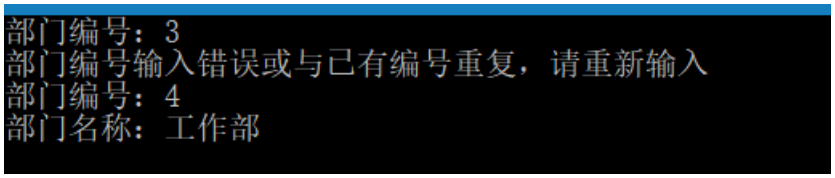
示例如下：“员工信息管理”系统的程序运行结果。

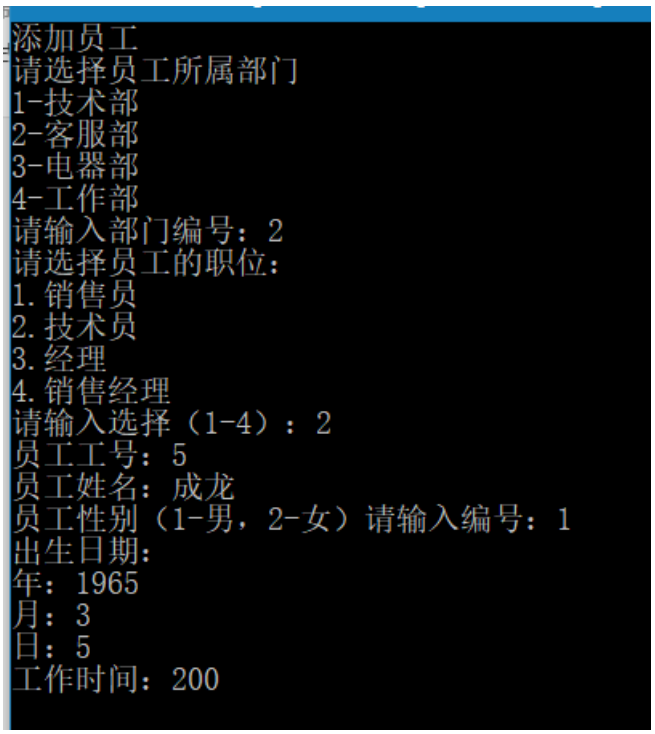
4.1 主控模块运行界面



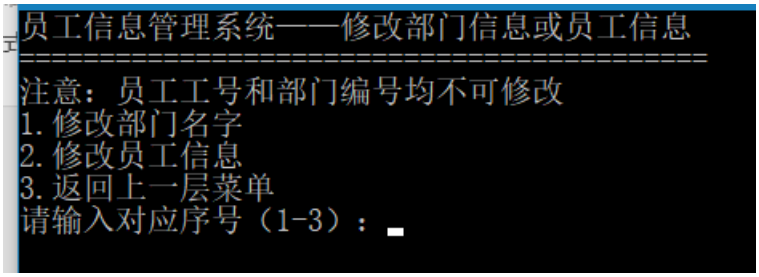
4.2 添加信息模块运行界面

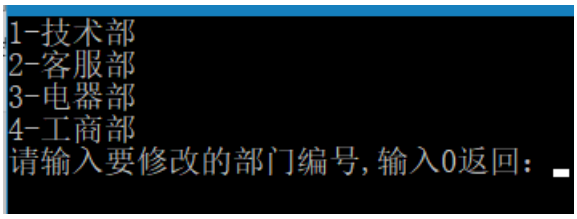


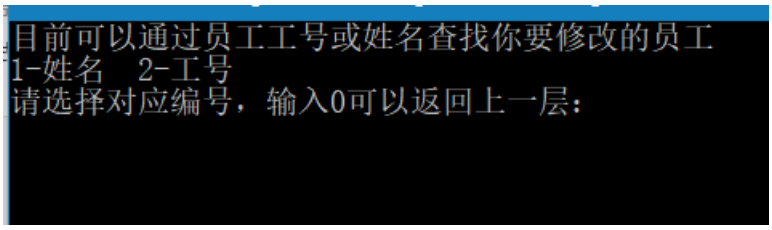




4.3 修改信息模块运行界面







```
员工号：5
姓名：成龙
性别：男
部门：2-客服部
出生日期:1965-3-5
职务：技术员
工作时间（小时）：200
工资：20000.00

选择要修改的项目
1-姓名  2-性别  3-出生日期  4-部门  5-岗位  6-工作时间
请选择（输入0返回）：
```

4.4 查询信息模块运行界面

```
员工信息管理系统——查询部门信息或员工信息
==========================================
1.查看已有部门
2.根据部门编号查询部门员工信息
3.根据部门名字查询部门员工信息
4.分页查询所有员工信息（按工资高低输出）
5.查询所有销售员信息
6.查询所有技术员信息
7.查询所有销售经理信息
8.查询所有经理信息
9.返回上一层菜单
请输入对应序号（1-9）：
```

```
1-技术部 人数：3
2-客服部 人数：3
3-电器部 人数：0
4-工商部 人数：0
请按任意键继续. . .
```

```
1-技术部 人数：3
工号   姓名   性别   部门    出生日期
  2     张三    男 1-技术部 1969-2-5 职务：技术员 工作时间（小时）：200 工资：20000.00
  1   肖肇斌    男 1-技术部 1997-2-2 职务：销售员 销售额：200000.00 工资：8000.00
  3     王五    男 1-技术部 1950-3-20 职务：销售经理 工资：6000.00
请按任意键继续. . .
```

```
共读取6条记录。
记录会按照工资从高到低排序，请输入你希望一页打印几条记录
输入一个大于0，小于或等于6的整数：3
工号   姓名   性别   部门    出生日期
 20   梁静茹   女 2-客服部 1980-5-3 职务：销售员 销售额：2000000.00 工资：80000.00
 21   林俊杰   男 2-客服部 1985-6-30 职务：技术员 工作时间（小时）：300 工资：30000.00
  2     张三   男 1-技术部 1969-2-5 职务：技术员 工作时间（小时）：200 工资：20000.00

该页已显示完全
1-下一页    2-返回
请选择（1-2）：
```

```
工号   姓名   性别   部门    出生日期
  5     成龙   男 2-客服部 1965-3-5 职务：技术员 工作时间（小时）：200 工资：20000.00
  1   肖肇斌   男 1-技术部 1997-2-2 职务：销售员 销售额：200000.00 工资：8000.00
  3     王五   男 1-技术部 1950-3-20 职务：销售经理 工资：6000.00

该页已显示完全
1-上一页    2-返回
请选择（1-2）：
```

```
工号   姓名   性别   部门    出生日期
  1   肖肇斌   男 1-技术部 1997-2-2 职务：销售员 销售额：200000.00 工资：8000.00
 20   梁静茹   女 2-客服部 1980-5-3 职务：销售员 销售额：2000000.00 工资：80000.00
========================
全部信息已显示完全
请按任意键继续. . .
```

4.5 删除信息模块运行界面

```
员工信息管理系统——删除部门信息或员工信息
========================================
注意：删除部门会同步删除该部门的员工信息
1.删除部门信息
2.删除员工信息
3.返回上一层
请输入对应序号（1-3）：
```

```
1-技术部 人数：3
工号   姓名   性别   部门    出生日期
  2     张三   男 1-技术部 1969-2-5 职务：技术员 工作时间（小时）：200 工资：20000.00
  1   肖肇斌   男 1-技术部 1997-2-2 职务：销售员 销售额：200000.00 工资：8000.00
  3     王五   男 1-技术部 1950-3-20 职务：销售经理 工资：6000.00
========================================
确定删除该部门及所有员工信息，请按1确定：
```

```
员工号：2
姓名：张三
性别：男
部门：1-技术部
出生日期:1969-2-5
职务：技术员
工作时间（小时）：200
工资：20000.00

确定删除该员工的信息，请按1确定：
```

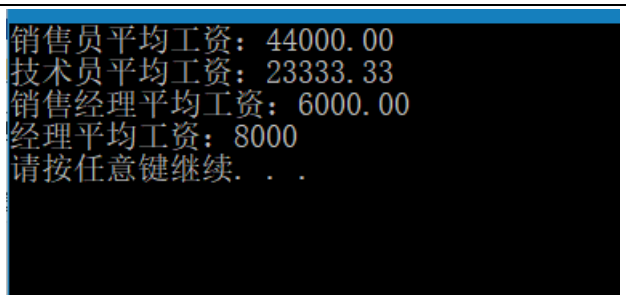
4.6 分析模块运行界面

```
员工信息管理系统——查询部门信息或员工信息
==========================================
1.统计并显示某个部门的平均月工资、最低月工资、最高月工资
2.统计并显示某个部门超出本部门平均月工资的人数与员工信息
3.统计并显示所有员工中的最低月工资和最高月工资员工的信息
4.统计并显示所有员工超出平均月工资的人数与员工信息
5.查看各岗位的平均工资
6.返回上一层菜单
请输入对应序号（1-6）：
```

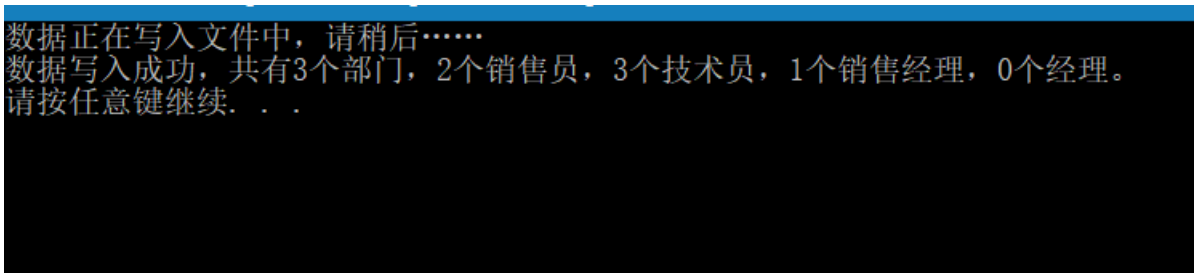
```
1-技术部 人数：3
平均工资：11333.33
最低工资：6000.00
最高工资：20000.00
请按任意键继续. . .
```

```
1-技术部 人数：3
工号  姓名  性别  部门  出生日期
  2    张三  男 1-技术部 1969-2-5 职务：技术员 工作时间（小时）：200 工资：20000.00
==========================================
全部信息已显示完全,超过平均工资的人数是1
请按任意键继续. . .
```

```
工号  姓名  性别  部门  出生日期
  2    张三  男 1-技术部 1969-2-5 职务：技术员 工作时间（小时）：200 工资：20000.00
 21  林俊杰  男 2-客服部 1985-6-30 职务：技术员 工作时间（小时）：300 工资：30000.00
  5    成龙  男 2-客服部 1965-2-13 职务：技术员 工作时间（小时）：200 工资：20000.00
 20  梁静茹  女 2-客服部 1980-5-3 职务：销售员 销售额：2000000.00 工资：80000.00
==========================================
全部信息已显示完全,超过平均工资的人数是4
请按任意键继续. . .
```



4.7 退出模块运行界面



5. 总结

5．1 自我评价及收获

要求：针对本人选择的课程设计题目，列出本人已经完成的项目，并对每项完成的任务进行自我评价，写出感想。在此基础上，还可对整个系统进行评价。（不小于 200 字）

这个 C++的课程设计用了大概一周左右的时间完成，在一开始是毫无头绪地在瞎打。由于没有一个明确的思路，不像学期初那个 C 语言的项目有明确的思路。我采用了面向对象（姑且这么叫吧），把每个对象，就是每个员工，每个部门可以做什么，怎么做想出来，想到什么就打什么，因为从一开始的思路就是正确的，中间也没有出现什么巨大的改动，所以这个项目就这么打完了。由此我深深体会到面向对象编程的好处。这个项目说不上难，但也算是一次小小的尝试，希望可以把它做得更好。

5．2 有待解决的问题及进一步完善的思路

要求：针对本人选择的课程设计题目，列出尚未实现的部分，并作相应的文字说明，同时给出进一步完成的思路。在此基础上，还可对整个系统进行评价。（不小于 200 字）

其实这个项目一开始就打算给它弄个 UI 界面的，在中间看过一些什么 MFC，什么 duilib 库，什么 QT 之类的，最后因为临近期末要复习其他科了，决定先放一放，然后就在控制台把它打完了。除此之外，由于当时分开多个头文件，发生了相互包含的问题，当时好像才写了 300 多行，瞬间爆了 300 多个错误，最后发现是相互 include 的问题，但是我并没有很好的解决方法，只能先把它们放在同一个头文件里面，确保只变异一次，这才解决了这个问题，其实也不能说是解决吧，毕竟如果对于大型点的项目，这样的方法本身就不安全，是不该被采用的。在没有更好的方法之前，只能表示“我也很绝望”。