犀利人事管理系统

目录

1. 系统需求分析

人事管理系统记录了员工的姓名、职务、等级和薪水的信息。设计此人事管理系统，提供了如下功能：

1) 录用新员工:在程序提示下，用户从键盘输入数据：首先根据提示输入员工职务编号，然后按照提示依次录入员工信息。如：

   Type：1 Name：Jason Grade：1

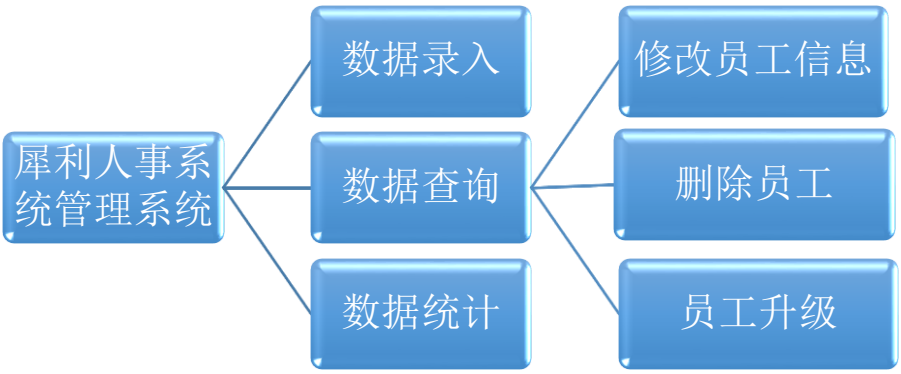
2) 查找员工:根据员工的姓名或 ID 准确查找员工，并打印员工的信息。
3) 修改员工信息：通过查找功能定位员工，修改员工的姓名等信息。
4) 解雇员工：删除员工信息（不可逆）。
5) 升级员工：将员工等级提升，同时更改员工的工资。
6) 统计员工信息：统计四种职务五种等级的员工的数量，打印到屏幕。统计所有员工的月付工资总和，打印到屏幕。

2. 总体设计

犀利人士管理系统包含 5 大功能，分别是：添加员工功能、按姓名或 ID 查找员工功能、修改员工信息功能、升职员工功能、统计员工信息功能。程序启动时将自动在目录下寻找”data.dat”。如果文件存在，则自动导入员工信息。若文件不存在，则创建文件。

程序的初始界面显示四个选项：

1. 数据录入
2. 数据查询
3. 数据统计
4. 退出



选择***数据录入***后会提示输入员工的种类，分别是经理、技术员、销售员、销售经理。选择种类过后，程序会要求与员工种类对应的信息，如：选择销售员会要求录入姓名与销售额，选择技术员会要求录入姓名、工作时间等。

选择***数据查询***后，可以通过员工 ID 和姓名两种方式进行查找。若没有找到与录入 ID 或姓名匹配的员工，系统将显示没有匹配项。若找到员工，程序将打印出员工信息，用户可继续对员工进行三项操作或返回。三项操作如下：

1. 更改信息
2. 删除员工
3. 升级员工

用户可选择*更改信息*更改员工的全部信息并保存；选择*删除员工*则员工信息

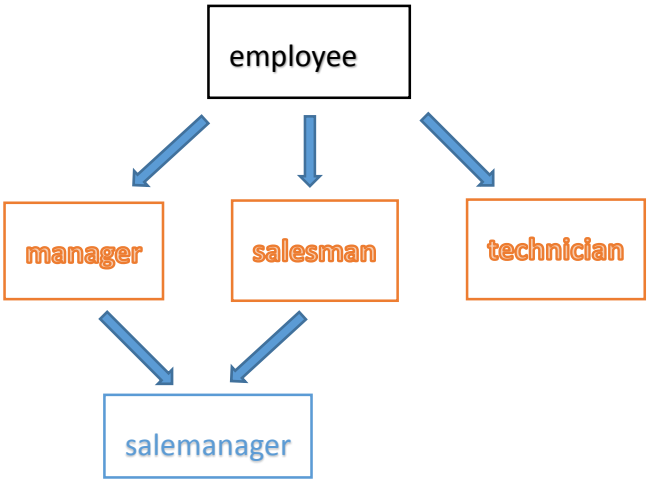
会被全部删除，不会保留在 data.dat 中；选择*升级员工*将员工等级+1，上限为 5。升级的同时系统将按特定的算法上调员工的工资。

选择***数据统计***后，程序将输出员工等级信息的统计，以及每月工资总和。

选择***退出***后，系统将把新添加的信息保存到文件中，并关闭窗口。

3. 详细设计

1. 类派生关系图



类派生关系说明：employee 类作为抽象类提供最基本的员工信息，其派生类在基础上增加工作时间、销售总额等内容。

2. 系统功能实现

(1) 创建新员工：程序会根据用户所选的员工类型自动创建相应的类对象；在用户录入员工姓名后进行一次检索，如果名字重复会提示用户重新键入；程序会自动为员工创建编号，编号会根据系统时间设定。

(2) 查找员工：程序提供两种精确查找方式，通过姓名和通过 ID 进行查找，查找到员工后程序打印出员工信息，如果未找到员工则显示”No Match Item”。

(3) 修改员工信息：在查找到员工后，搜索函数会返回员工的数组编号，可调用员工类的函数进行三项操作：修改员工姓名等信息、删除员工(不可逆)、将员工升级。

(4) 统计员工信息：统计四种职务五种等级的员工的数量，打印到屏幕。统计所有员工的月付工资总和，打印到屏幕。

4. 系统调试

在调试的过程中主要遇到一下几个问题：

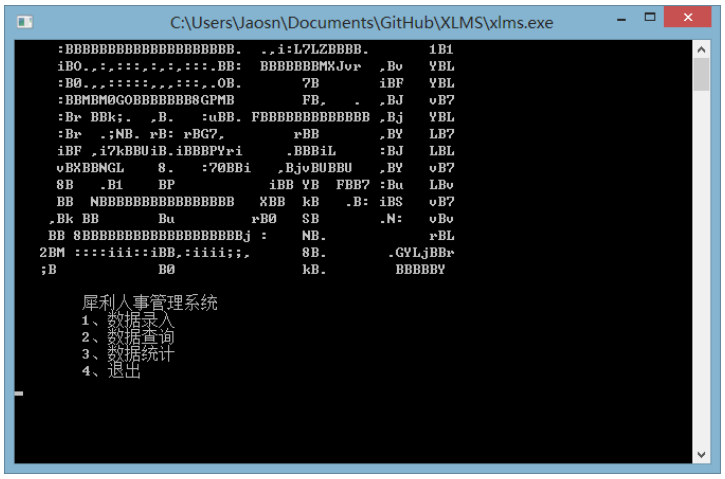
1） 一开始直接将对象直接保存到二进制文件中。在读取的时候发现总是各种各样的错误。后来经过助教的解答知道是对象中的函数指针在每次运行的时候改变造成的。于是增加了结构体，在类中增加了通过结构体构造类的构造函数和返回结构体的函数。在对象与文件中加入结构体作为数据传递的途径解决了这一问题。

2） 如何在二进制文件中删除已存在的条目？我想过两个解决方案：一个是将删除后的对象输出到一个新的文件如 data.dat.tmp 然后用 system 命令删除原来的 data.dat 再把新文件重命名；或是将文件里的内容读到内存里，然后清空原文件，在写入的过程中跳过要删除的条目。

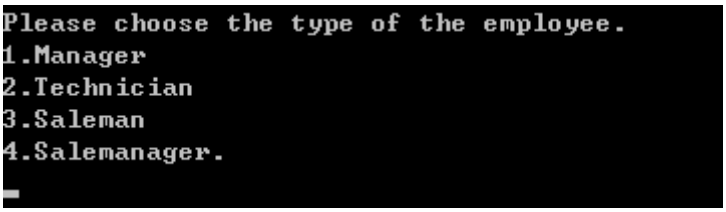
3） 如何让退回页面后重新显示菜单？我一开始想过用 while 循环输出菜单（就像单片机那样），和 goto 函数。后来在提示输入错误让用户重新输入的地方找到了灵感。于是让菜单递归自己，结合清屏函数达到了预想的效果。

4） 如何解决 vs 编译器 LINK2019 错误。这个错误往往是类里面某个函数（最有可能的是构造函数和析构函数）没有定义。换成 gcc 后会提示哪里哪里有问题，就好解决了(这也导致后面写的代码中全是英文)。

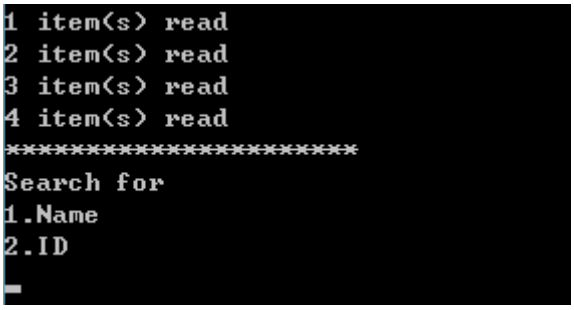
5. 测试结果与分析



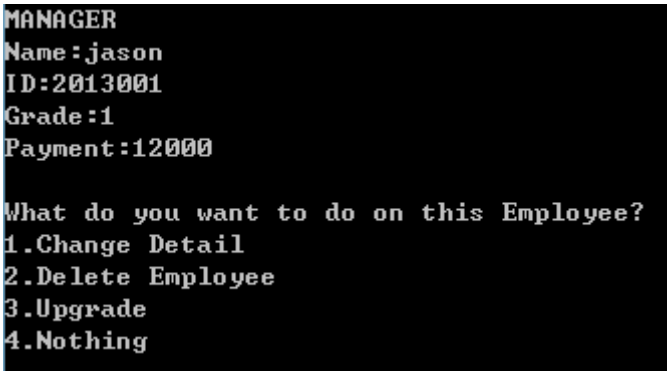
主界面



选择员工职位界面



查询员工界面

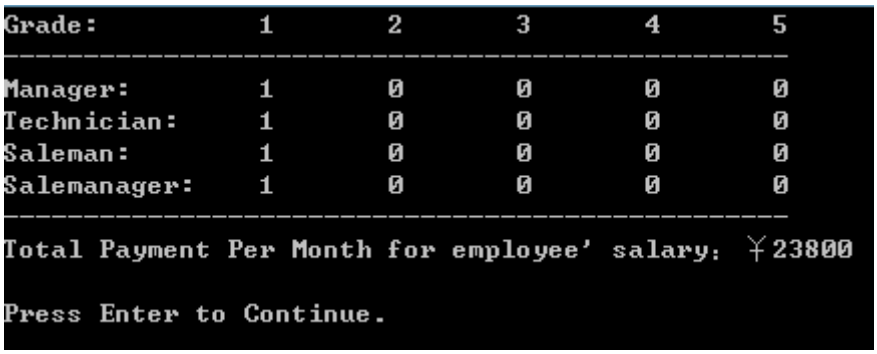


查询结果

数据统计界面

这一作业是对上一学期的犀利哥作业的整理补充。作业中很多代码再看的时候已经忘了当初写它的目的。由于程序的整体结构已经定死，这次的工作只是调整代码格式，删去无用的函数。同时增加了结构体，改变了原来对象到文件的读写方式，通过增加结构体避免了错误。这一方法在后面的作业中得到沿用。

总体感觉在这一作业中，限于原有的成果，没能写出有创新性的功能。同时，



自己感觉在这一作业中，并没有领会***面向对象编程***的精神。写的类就像是带有 private 属性的结构体，毫无生气。只当是对 debug 能力的提升。（在写完第一个作业后，看了另一门面向对象的语言，发现了类的灵活性，在第二个作业中争取将面向对象编程的精神体现出来。）

附：程序源代码

employee.h

```
#ifndef EMPLOYEE
#include<iostream>
#include<cstring>
#include<cstdlib>
#include<fstream>
#include<time.h>
using namespace std;
/*----------------------------
由于原版本是按照课程要求一点点更改的
其中许多函数都是多余的，如构造函数等等
因此在这个版本中，在实际过程中用不到的构造函数予以删除
```

```
同样的，作为友元的运算符重载，在程序中一次都没用到
但念想到当初敲那么多不容易，以后的后续功能加入可以调用
便予以保留
----------------------------*/
//设定编号起始、小时工资、提成比例、枚举型
char *rout= "data.dat";
int em_sum,em_num;     //员工编号为年份+三位如2013001
const int hour_pay=260;          //设定每小时工资
const double pay_percent=0.05;       //设定提成比例
enum type{_manager,_technican,_saleman,_salemanager};
//设定存储的结构体内容
struct data
{
    enum type _type;
    char name[24];
    int id;
    int grade;
    float accumPay;
    float work_hour;
    float sale_sum;
};
//以下为类定义
class employee    //虚基类删去了无用的构造函数
{
protected:
    char name[10];
    int individualEmpNo;
    int grade;
    double accumPay;
public:
    employee(){}
    virtual ~employee(){}
    virtual data transport()=0;
    virtual void upgrade() =0;
    friend istream& operator>>(istream&,employee&);
    friend ostream& operator<<(ostream&,employee&);
    virtual bool search(int id)=0;
    virtual bool search(char *name)=0;
    virtual void change_detail(int num)=0;
};
class manager: virtual public employee
{
    public:
        manager():employee(){}
```

```
        manager(data p);
        void upgrade();
        bool search(int id);
        bool search(char *name);
        data transport();
        void change_detail(int);
        friend istream& operator>>(istream&,manager&);
        friend ostream& operator<<(ostream&,manager&);
};
class technician : public employee
{
    public:
        technician():employee(){}
        technician(data p);
        void upgrade();
        bool search(int id);
        bool search(char *name);
        data transport();
        void change_detail(int);
        friend istream& operator>>(istream&,technician&);
        friend ostream& operator<<(ostream&,technician&);
    private:
        int work_hour;
};
class saleman: virtual public employee
{
    public:
        saleman():employee(){}
        saleman(data p);
        void upgrade();
        bool search(int id);
        bool search(char *name);
        data transport();
        void change_detail(int);
        friend istream& operator>>(istream&,saleman&);
        friend ostream& operator<<(ostream&,saleman&);
    protected:
        int sale_sum;
};
class salemanager:public saleman,public manager
{
    public:
        salemanager():manager(),saleman(){}
        salemanager(data p);
```

```
        void upgrade();
        bool search(int id);
        bool search(char *name);
        data transport();
        void change_detail(int);
        friend istream& operator>>(istream&,salemanager&);
        friend ostream& operator<<(ostream&,salemanager&);
};
bool check_name(char* name)
{
    fstream check(rout,ios::binary|ios::in);
    data temp;
    bool flag=0;
    for(int i=0;i<em_sum;i++)
    {
        check.read((char*)&temp,sizeof(data));
        if(strcmp(name,temp.name)==0) flag=1;
    }
    check.close();
    return flag;
}
bool check_name(char* name,int num)
{
    fstream check(rout,ios::binary|ios::in);
    data temp;
    bool flag=0;
    for(int i=0;i<em_sum;i++)
    {
        check.read((char*)&temp,sizeof(data));
        if(strcmp(name,temp.name)==0&&i!=num) flag=1;
    }
    check.close();
    return flag;
}
istream& operator >>(istream& cin,employee & temp)
{
    cout<<"Please input Grade:";
    cin>>temp.grade;
    cout<<"Please input Payment:";
    cin>>temp.accumPay;
    return cin;
}
ostream& operator <<(ostream& cout,employee & temp)
{
```

```
    cout<<temp.individualEmpNo<<endl<<temp.grade<<endl<<temp.accumPay<<endl<<endl;
    //id,grade,accumpay
    return cout;
}
istream& operator >>(istream& cin,manager & temp)              //重载各种输入输出
{
    cout<<"Please input Grade:";
    cin>>temp.grade;
    return cin;
}
ostream& operator <<(ostream& cout,manager & temp)
{
    cout<<"NO:"<<temp.individualEmpNo<<endl<<"Grade"<<temp.grade<<endl<<"Payment:"<<tem
p.accumPay<<endl<<endl;
    return cout;
}
istream& operator >>(istream& cin,technician & temp)
{
    cout<<"Please input Grade:";
    cin>>temp.grade;
    cout<<"Please input Work Hours:";
    cin>>temp.work_hour;
    return cin;
}
ostream& operator <<(ostream& cout,technician & temp)
{
    cout<<"NO:"<<temp.individualEmpNo<<endl<<"Grade"<<temp.grade<<endl<<"Payment:"<<tem
p.accumPay<<endl<<endl;
    return cout;
}
istream& operator >>(istream& cin,saleman & temp)
{
    cout<<"Please input Grade:";
    cin>>temp.grade;
    cout<<"Please input Saleroom:";
    cin>>temp.sale_sum;
    return cin;
}
ostream& operator <<(ostream& cout,saleman & temp)
{
    cout<<"NO:"<<temp.individualEmpNo<<endl<<"Grade"<<temp.grade<<endl<<"Payment:"<<tem
p.accumPay<<endl<<endl;
    return cout;
}
```

```
istream& operator >>(istream& cin,salemanager & temp)
{
    cout<<"Please input Grade:";
    cin>>temp.grade;
    cout<<"Please input Saleroom:";
    cin>>temp.sale_sum;
    return cin;
}
ostream& operator <<(ostream& cout,salemanager & temp)
{
    cout<<"NO:"<<temp.individualEmpNo<<endl<<"Grade"<<temp.grade<<endl<<"Payment:"<<tem
p.accumPay<<endl<<endl;
    return cout;
}
//--------------------
//以下为搜索函数
bool manager::search(int id)
{
    if (individualEmpNo==id)
    {
        cout<<"MANAGER\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<id<<endl
            <<"Grade:"<<grade<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool manager::search(char *name)
{
    if (strcmp(this->name,name)==0)
    {
        cout<<"MANAGER\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<individualEmpNo<<endl
            <<"Grade:"<<grade<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool technician::search(int id)
{
```

```
    if (individualEmpNo==id)
    {
        cout<<"TECHNICIAN\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<id<<endl
            <<"Grade:"<<grade<<endl
            <<"Workhour:"<<work_hour<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool technician::search(char *name)
{
    if (strcmp(this->name,name)==0)
    {
        cout<<"TECHNICIAN\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<individualEmpNo<<endl
            <<"Grade:"<<grade<<endl
            <<"Workhour:"<<work_hour<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool saleman::search(int id)
{
    if (individualEmpNo==id)
    {
        cout<<"SALEMAN\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<id<<endl
            <<"Grade:"<<grade<<endl
            <<"Salesum"<<sale_sum<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool saleman::search(char *name)
{
    if (strcmp(this->name,name)==0)
    {
```

```
        cout<<"SALEMAN\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<individualEmpNo<<endl
            <<"Grade:"<<grade<<endl
            <<"Salesum:"<<sale_sum<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool salemanager::search(int id)
{
    if (individualEmpNo==id)
    {
        cout<<"SALEMANAGER\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<id<<endl
            <<"Grade:"<<grade<<endl
            <<"Salesum"<<sale_sum<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
bool salemanager::search(char *name)
{
    if (strcmp(this->name,name)==0)
    {
        cout<<"SALEMANAGER\n"
            <<"Name:"<<name<<endl
            <<"ID:"<<individualEmpNo<<endl
            <<"Grade:"<<grade<<endl
            <<"Salesum:"<<sale_sum<<endl
            <<"Payment:"<<accumPay<<endl;
        return 1;
    }
    else return 0;
}
//------------
//以下为依靠结构体构建类函数
manager::manager(data p)
{
    strcpy(name,p.name);
    individualEmpNo=p.id;
```

```
    grade=p.grade;
    accumPay=p.accumPay;
    if(p.id>em_sum) em_num=p.id;
}
technician::technician(data p)
{
    strcpy(name,p.name);
    individualEmpNo=p.id;
    grade=p.grade;
    work_hour=p.work_hour;
    accumPay=p.accumPay;
    if(p.id>em_sum) em_num=p.id;
}
saleman::saleman(data p)
{
    strcpy(name,p.name);
    individualEmpNo=p.id;
    grade=p.grade;
    sale_sum=p.sale_sum;
    accumPay=p.accumPay;
    if(p.id>em_sum) em_num=p.id;
}
salemanager::salemanager(data p)
{
    strcpy(name,p.name);
    individualEmpNo=p.id;
    grade=p.grade;
    sale_sum=p.sale_sum;
    accumPay=p.accumPay;
    if(p.id>em_sum) em_num=p.id;
}
//--------------
//以下为升级函数
void manager::upgrade()
{
    if(grade<5)
    {
        grade++;
        accumPay++;
        accumPay=accumPay*1.1;
    }
    else
    {
        cout<<"Can't upgrade. \nCurrent Grade is 5!";
```

```
            getchar();
        }
}
void technician::upgrade()
{
    if(grade<5)
    {
        grade++;
        accumPay*=1.2;
    }
    else
    {
        cout<<"Can't upgrade. \nCurrent Grade is 5!";
        getchar();
    }
}
void saleman::upgrade()
{
    if(grade<5)
    {
        grade++;
        accumPay*=1.2;
    }
    else
    {
        cout<<"Can't upgrade. \nCurrent Grade is 5!";
        getchar();
    }
}
void salemanager::upgrade()
{
    if(grade<5)
    {
        grade++;
        accumPay=1.1*(accumPay-sale_sum*pay_percent)+sale_sum*pay_percent*1.2;
    }
    else
    {
        cout<<"Can't upgrade. \nCurrent Grade is 5!";
        getchar();
    }
}
//--------------------
//以下为类生成结构体的函数
```

```
data manager::transport()
{
    data temp;
    temp._type=(type)0;
    temp.accumPay=accumPay;
    temp.grade=grade;
    temp.id=individualEmpNo;
    temp.sale_sum=0;
    temp.work_hour=0;
    strcpy(temp.name,name);
    return temp;
}
data technician::transport()
{
    data temp;
    temp._type=(type)1;
    temp.accumPay=accumPay;
    temp.grade=grade;
    temp.id=individualEmpNo;
    temp.sale_sum=0;
    temp.work_hour=work_hour;
    strcpy(temp.name,name);
    return temp;
}
data saleman::transport()
{
    data temp;
    temp._type=(type)2;
    temp.accumPay=accumPay;
    temp.grade=grade;
    temp.id=individualEmpNo;
    temp.sale_sum=sale_sum;
    temp.work_hour=0;
    strcpy(temp.name,name);
    return temp;
}
data salemanager::transport()
{
    data temp;
    temp._type=(type)3;
    temp.accumPay=accumPay;
    temp.grade=grade;
    temp.id=individualEmpNo;
    temp.sale_sum=sale_sum;
```

```
    temp.work_hour=0;
    strcpy(temp.name,name);
    return temp;
}
//----------------
//以下为更改对象数据的函数
void manager::change_detail(int num)
{
    char temp[24];
    cout<<"Input New Name:" ;
    cin>>temp;
    if(!check_name(temp,num))strcpy(name,temp);
    else
        cout<<"Name exits! You need to retype a new name\nLike: \n\tif Jason
exist\n\ttry Jason(1)";
}
void technician::change_detail(int num)
{
    char temp[24];
    cout<<"Input New Name:" ;
    cin>>temp;
    if(!check_name(temp,num))
    {
        strcpy(name,temp);
        cout<<"Input New Workhour:" ;
        cin>>work_hour;
        accumPay=work_hour*hour_pay;
    }
    else
        cout<<"Name exits! You need to retype a new name\nLike: \n\tif Jason
exist\n\ttry Jason(1)";

}
void saleman::change_detail(int num)
{
    char temp[24];
    cout<<"Input New Name:" ;
    cin>>temp;
    if(!check_name(temp,num))
    {
        strcpy(name,temp);
        cout<<"Input New Salesum:" ;
        cin>>sale_sum;
        accumPay=sale_sum*pay_percent;
```

```
    }
    else
        cout<<"Name exits! You need to retype a new name\nLike: \n\tif Jason
exist\n\ttry Jason(1)";

}
void salemanager::change_detail(int num)
{
    char temp[24];
    cout<<"Input New Name:" ;
    cin>>temp;
    if(!check_name(temp,num))
    {
        strcpy(name,temp);
        cout<<"Input New Salesum:" ;
        cin>>sale_sum;
        accumPay=8000+sale_sum*pay_percent*0.8;
    }
    else
        cout<<"Name exits! You need to retype a new name\nLike: \n\tif Jason
exist\n\ttry Jason(1)";

}

#define EMPLOYEE
#endif
```

xlms.cpp

```
#include"employee.h"
void init()
{
  time_t t;
    t=time(NULL);
    em_num=1000*(t/31104000+1969);
    fstream infile(rout,ios::in|ios::out|ios::binary|ios::ate);
    em_sum=infile.tellg()/sizeof(data);
    infile.seekp((em_sum-1)*sizeof(data),ios::beg);  //寻找最大编号
    data temp;
    infile.read((char *)&temp,sizeof(data));
    if(temp.id>em_num) em_num=temp.id;
    infile.close();
}
void gen()
```

```
{
    data temp;
    int grade_static[4][6]={0};
    int pay_sum=0;
    fstream in(rout,ios::in|ios::binary);
    for(int i=0;i<em_sum;i++)
    {
        in.read((char*)&temp,sizeof(data));
        grade_static[temp._type][temp.grade-1]++;
        pay_sum+=temp.accumPay;
    }
    in.close();
    system("cls");
    cout<<"Grade:\t\t1\t2\t3\t4\t5\n-------------------------------------------------
\nManager:";
    for(int i=0;i<5;i++) cout<<"\t"<<grade_static[0][i];
    cout<<"\nTechnician:";
    for(int i=0;i<5;i++) cout<<"\t"<<grade_static[1][i];
    cout<<"\nSaleman:";
    for(int i=0;i<5;i++) cout<<"\t"<<grade_static[2][i];
    cout<<"\nSalemanager:";
    for(int i=0;i<5;i++) cout<<"\t"<<grade_static[1][i];
    cout<<"\n-------------------------------------------------\nTotal Payment Per Month
for employee' salary: ￥"<<pay_sum
        <<"\n\nPress Enter to Continue.\n";
    getchar();getchar();
}
void addinfo()
{
    system("cls");
    int ch;
    data temp;
    cout<<"Please choose the type of the
employee.\n1.Manager\n2.Technician\n3.Saleman\n4.Salemanager.\n";
    cin>>ch;
    temp._type=(type)(ch-1);                    //设定种类
    switch(ch)
    {
        case 1:                          //输入manager类
            temp.accumPay=12000; //系统设定
            temp.id=++em_num;
            cout<<"Input Name:" ;
            cin>>temp.name;
            temp.grade=1;
```

```
            break;
        case 2:
            temp.id=++em_num;
            cout<<"Input Name:" ;
            cin>>temp.name;
            temp.grade=1;
            cout<<"Input Workhour:";
            cin>>temp.work_hour;
            temp.accumPay=temp.work_hour*hour_pay;
            break;
        case 3:
            temp.id=++em_num;
            cout<<"Input Name:" ;
            cin>>temp.name;
            temp.grade=1;
            cout<<"Input Salesum:";
            cin>>temp.sale_sum;
            temp.accumPay=temp.sale_sum*pay_percent;
            break;
        case 4:
            temp.id=++em_num;
            cout<<"Input Name:" ;
            cin>>temp.name;
            temp.grade=1;
            cout<<"Input Salesum:";
            cin>>temp.sale_sum;
            temp.accumPay=8000+temp.sale_sum*pay_percent*0.8;
            break;
        default:
            cout<<"Wrong input! Try again.\n"
            <<"\nPress Enter to Continue.\n";
            getchar();getchar();       //提供停顿的老办法
            addinfo();return;                  //递归重新输入
    }
    for(int i=0;i<em_sum;i++)
    {

    }
    if(check_name(temp.name))
    {
        cout<<"Name exits! You need to retype a new name\nLike: \n\tif Jason
exist\n\ttry Jason(1)";
        getchar();getchar();
        addinfo();
```

```
        return;
    }
    ofstream outfile;
    outfile.open(rout,ios::binary|ios::out|ios::app);
    outfile.write((char *)&temp,sizeof(data));
    outfile.close();
    em_sum++;
}
void readwrite()
{
    system("cls");        //清个屏
    int wrong_int;
    int count=0;
    int flag=-1;
    data temp;            //建立临时读取结构体
    employee **temp_em = new employee*[em_sum];      //为全体员工创建数组
    fstream infile(rout,ios::in|ios::out|ios::binary);
    if(!infile)
    {
        throw wrong_int;
        return;
    }

    infile.seekg(ios::beg);
    //下面读入了文件中的所有数据
    for(int i=0;i<em_sum;i++)
    {
        infile.read ((char*)&temp,sizeof(data));
        switch(temp._type)
        {
            case 0:
                temp_em[i]=new manager(temp);
                break;
            case 1:
                temp_em[i]=new technician(temp);
                break;
            case 2:
                temp_em[i]=new saleman(temp);
                break;
            case 3:
                temp_em[i]=new salemanager(temp);
                break;
        }
        cout<<i+1<<" item(s) read"<<endl;
```

```
    }
    infile.close();
    //下面开始搜索
    cout<<"**********\nSearch for\n1.Name\n2.ID\n";
    cin>>wrong_int;
    switch(wrong_int)
    {
    case 1:
        system("cls");
        cout<<"Name:\n";
        cin>>temp.name;
        system("cls");
        for(int i=0;i<em_sum;i++) count+=(1+i)*temp_em[i]->search(temp.name);
        break;
    case 2:
        system("cls");
        cout<<"ID:\n";
        cin>>temp.id;
        system("cls");
        for(int i=0;i<em_sum;i++) count+=(i+1)*temp_em[i]->search(temp.id);
        break;
    default:
        system("cls");
        cout<<"Wrong Input\n";
        break;
    }
    if (count==0)
    {
        cout<<"No Match Item\n";
        getchar();
    }
    else
    {
        count--;
        cout<<"\nWhat do you want to do on this Employee?\n1.Change Detail\n2.Delete
Employee\n3.Upgrade\n4.Nothing\n";
        cin>>wrong_int;
        switch(wrong_int)
        {
            case 1:
                temp_em[count]->change_detail(count);
                break;
            case 2:
                flag=count;
```

```
                break;
            case 3:
                temp_em[count]->upgrade();
                break;
            case 4:
                break;
            default:
                cout<<"Wrong Input\n"
                <<"\nPress Enter to Continue.\n";
                getchar();getchar();
        }
        fstream change(rout,ios::out|ios::trunc|ios::binary);
        for(int i=0;i<em_sum;i++)
        {
            if(i!=flag)
            {   temp=temp_em[i]->transport();
                change.write((char *)&temp,sizeof(data));
            }
        }
        change.close();
        init();
    }
    for(int i=0;i<em_sum;i++) delete temp_em[i];
    delete [] temp_em;
    cout<<"\nPress Enter to Continue.\n";
    getchar();
}
void printscreen()
{
    int chose_temp;
    system("cls");
    cout<<"     :BBBBBBBBBBBBBBBBBBBB.  .,i:L7LZBBBB.        1B1        \n
iBO.,:,:::,:,:,:::.BB:  BBBBBBBMXJvr  ,Bv   YBL        \n     :BO.,,:::::,,,:::,.OB.
7B       iBF   YBL        \n     :BBMBMOGOBBBBBBB8GPMB        FB,   .  ,BJ   vB7
\n     :Br BBk;.  ,B.   :uBB. FBBBBBBBBBBBBB ,Bj   YBL        \n     :Br  .;NB. rB:
rBG7,        rBB       ,BY   LB7        \n
iBF ,i7kBBUiB.iBBBPYri     .BBBiL     :BJ   LBL        \n     vBXBBNGL
8.   :70BBi   ,BjvBUBBU   ,BY   vB7        \n     8B   .B1    BP           iBB YB
FBB7 :Bu   LBv        \n     BB  NBBBBBBBBBBBBBBB   XBB  kB   .B: iBS   vB7
\n    ,Bk BB       Bu         rBO   SB       .N:   vBv        \n    BB
8BBBBBBBBBBBBBBBBBBBj :    NB.            rBL        \n   2BM ::::iii::iBB,:iiii;:,
8B.       .GYLjBBr        \n   ;B            BO               kB.        BBBBBY
\n";
    cout<<"\n\t犀利人事管理系统\n\t1、数据录入\n\t2、数据查询\n\t3、数据统计\n\t4、退出
```

```
\n";
    cin>>chose_temp;
    try
    {
        switch(chose_temp)
        {
            case 1:
                addinfo();
                printscreen();
                break;
            case 2:
                readwrite();
                printscreen();
                break;
            case 3:
                gen();
                printscreen();
            case 4:
                return;
                break;
            default: break;
        }
    }
    catch(int)
    {
        cout<<"File does not exist!"<<endl
            <<"\nPress Enter to Continue.\n";
        getchar(); getchar();
        printscreen();
    }
}
int main()
{
    init();
    printscreen();
    system("cls");
    cout<<"BYEBYE"<<endl;
    return 0;
}
```