《企业工资管理系统》开发文档

一、程序功能概述

本程序是一款企业员工工资管理软件。系统内置 3 种人员：工人、技术人员、企业人员。每个人的工资由两个部分组成：底薪和奖金。底薪的值由系统内置（工人 3000 元，技术人员 7000 元，管理人员 12000 元）。本程序可以实现：录入员工信息（工号、姓名、性别、人员类别、手机、奖金）、修改员工信息、删除员工、给员工发奖金、给员工扣奖金、搜索符合给定条件的员工（仅支持逻辑与）、统计员工工资信息（包括“所有员工”和“符合给定条件的员工”两组信息）、将员工信息存储为二进制文件进行保存、读取。

二、开发环境与配置要求

本程序全部使用 DEV-C++开发，建议在 Window 7/8/10 下运行。不需要 Qt、miniSQL 等软件支持。

若需查看源码，可在安装了 DEV-C++的前提下，双击 Manager.dev。否则，请自行将代码加入你所使用的编译器的同一工程内。

三、总体架构设计

本程序的主要操作对象是员工信息。而本程序中，所有员工信息都被存放在一条链表（EmpList 类）内。程序的主要功能，如发奖金、插入、删除、更新人员信息等，也是由一个活动的指针直接操纵着这条链表上的数据。通过遍历这一条链表，我们可以实现工资信息统计。

这一条链表（可称作 all）还拥有一条子链（可称作 show）。这条子链并不一定链住所有的员工信息，但它链住了所有符合用户搜索条件的员工信息。也就是说，这条子链被用与于用户交互，即搜索区的输入可以操纵这条子链，这条子链进而被用于显示符合条件的搜索结果。

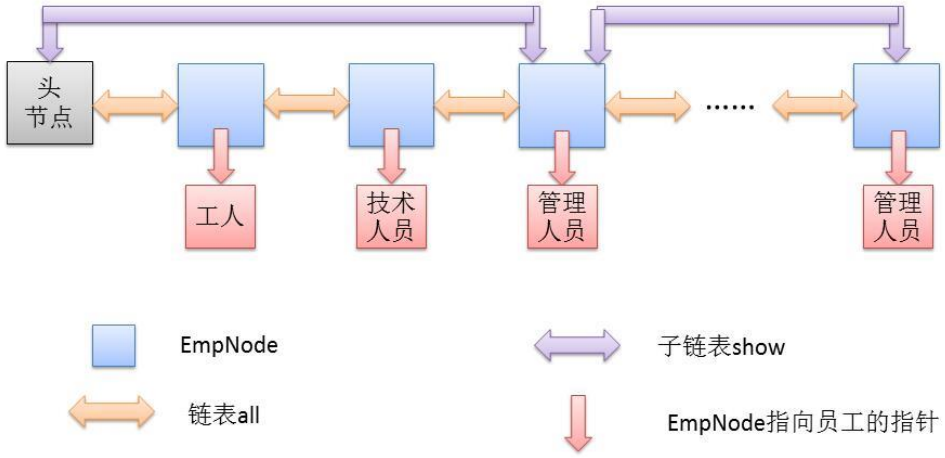

图 1 链表结构示意

程序界面包括三部分：搜索区、员工信息管理区、统计区。界面接受的信息可能来自于：搜索区、员工信息管理区；界面反馈的信息可能显示在：员工信息管理区、统计区。用户通过多个按钮、单行文本编辑框，让界面模块接收用户的交互信息。界面模块接着调用自己或其他类的相关方法进行实际的数据处理，然后将处理结果显示在界面上，以反馈给用户。

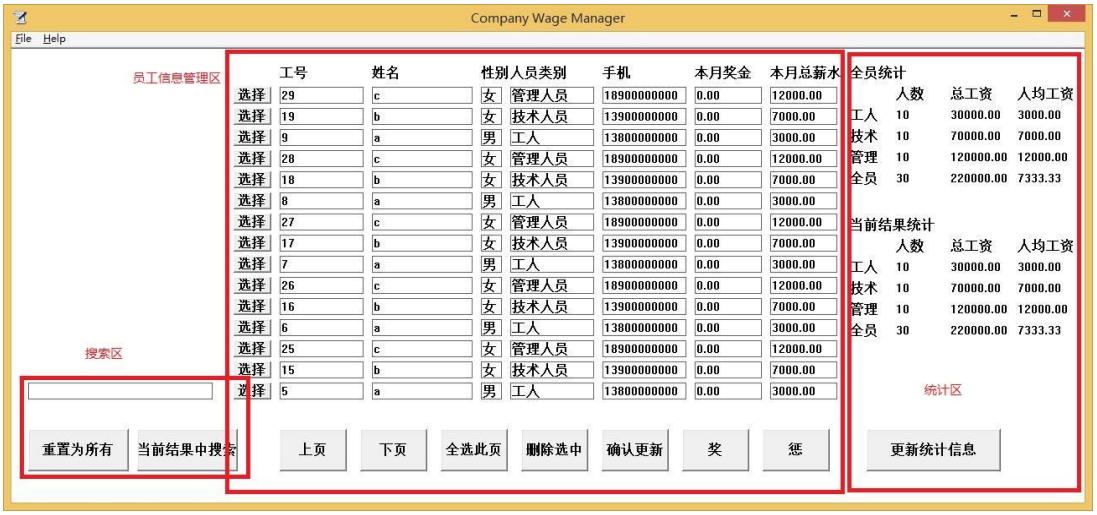

图 2 界面示意

界面上还提供了二进制文件的存取功能，实现了员工数据的离线存储、转移。

四、模块功能设计与说明

（一）员工类设计

纯虚基类 Employee，继承类 Worker、Tech、Manager。

每个继承类从基类继承 5 个成员变量，分别为工号 ID、姓名 name、性别 sex（1 为男，0 为女）、手机 cell、工资 wage。其中工资是另一个自定义类 Wage。注意，每个继承类的大

小都是 48B。这个数据对其他模块的运行很重要。

```
protected:
    long long ID;
    char name[23];
    char sex;
    long long cell;
    Wage wage;
```

图 3 emp.h 中 Employee 类的成员

每个继承类有另外的一个静态成员变量 count，构造时自增，析构时自减，用于记录此类员工共有多少个。这个信息的记录可以为保存为二进制文件时提供便利（计算需要占用的空间）。

纯虚函数有：析构函数、奖惩函数即 Reward()和 Punish()。将奖惩函数设计为虚函数的原因是，职位不同的人，违规或立功所承受的奖金变化是不同的，比如普通工人迟到扣 400 元，技术员工迟到扣 1000 元，管理人员迟到扣 1800 元。

关于这些人员类的其他设计，都是些简单的 set/get 的封装、构造函数，故略去。

上面提到薪水类 Wage。它也比较简单，有两个成员，base 和 bonus，其中 base 值对于每种员工而言是固定的。

（二）员工信息存储结构及相关操作

1. EmpNode

程序中，每一个员工对象实际放在节点类 EmpNode 中管理。

EmpNode 成员：

1. 四个 EmpNode*指针（lall, nall, lshow, nshow），分别指向：all 链表的前一个节点、all 链表的后一个节点、show 链表的前一个节点、show 链表的后一个节点。
2. 一个 Employee*指针(data)，指向这个节点所管理的那一个员工实例。

EmpNode 方法（注：本报告在描述方法时，可能省略它的输入，也就是说看到 f()这样的表示未必代表 f 这个函数无参）：

1. 五个 Get 分别用于获取上述五个指针指向的地址。
2. 以 48B 为输入的方法有两个：一个是构造函数，即根据输入的 48B 生成一个新员工，放到这个 EmpNode 内；另一个是 Refresh()，这个方法用在更新员工信息的时候，即根据用户提供的最新的那 48B，来改造它原本指向的那个员工。实际实现的方法是，先将 data 指针 delete 掉，再根据 48B new 一个 Employee 的继承类。是哪个继承类，也取决于 48B 内的信息。

48B 与员工对象之间的转换是很简单的，本应略去，但是由于它在本报告后续中不断出现，比较关键，因此还是给出可以体现转换过程的一小部分代码，以帮助理解。请重点关注加粗、下划线的两句，以了解 48B 内的信息结构。实际上，这 48B 包含的信息顺序和图 3 的成员顺序一致。Wage 的 bonus 不用设置，因为这根据员工类型确定：

```
EmpNode::EmpNode(const char* txt48B ){
```

```
char name[30];
int i;
for(i=0;i<23;i++){
    name[i] = txt48B[i+8];
}
name[i] = 0;
this->lall = this->lshow = this->nall = this->nshow = 0;
char type = (txt48B[31]>>6) & 3 ;
if(type==0){
    this->data = new Worker(*(long long*)txt48B, name, txt48B[31],
*(long long*)(txt48B+32));
    this->data->SetWageBonus(*(float*)(txt48B + 44));
}
... ...
}
```

2. EmpList

程序中，存在且仅存在一个 EmpList。

EmpList 成员：

1. Emplist 只有一个成员，就是一个 EmpNode，变量名为 head。它是一个空节点。它的 lall、lshow 和 data 都是空指针。它的 nall 指向第一个员工对象、nshow 指向第一个符合用户搜索条件的员工对象。

EmpList 方法：

1. 两个 Get 函数，分别取得 nall 和 nshow 指针的指向地址。这使得完整遍历链表成为可能。

2. SetNodes()和 GetNodes()。这两个函数是用来实现链表中的所有节点（也就是 all 链）与 char*变量的互化。实际上就是为了实现二进制文件的存取。这个操作以 48B 为一个单元进行。

3. 两个 Insert()（其一输入为 Employee*，其二输入为 EmpNode*，无本质差别，只是一个 new EmpNode 的差异）、一个 Delete()（输入为 EmpNode*），它们的操作很类似，都只是修改指针指向。注意 EmpNode 的 5 个指针成员都要涉及到。

插入时，插在 head 节点后方，即链表头。假设原本 head 的 nall 指向 X，nshow 指向 Y，则插入操作将把双向的 all 指针、show 指针都修改掉。具体地说，设新插入的节点为 N。则 head 的 nall、nshow 均指向 N；X（若非空）的 lall 指向 N，Y（若非空）的 lshow 指向 N。

删除时，设输入的指针指向 D，则 D 的 lall 的 nall 指向 D 的 nall；D 的 nall（若非空）的 lall 指向 D 的 lall。类似地处理 show 那套链表。

4. Clear()。这个函数用于删除 EmpList 内的所有数据，包括所有 EmpNode 指向的人员对象。

5. Show()。这个函数用于“重置为所有”按钮功能的实现。它使得 EmpList 中每个节点的 nshow 与 nall 相等、lshow 与 lall 相等。

6. SelectShow()。这个函数用于实现员工搜索。它接收一个 char*，然后调用 interpreter 类的 translate()和 interpret()来完成条件选择。这两个函数请参见下一节。

（三）搜索功能实现与搜索语法说明

1. 翻译类 interpreter 设定

设置翻译类，class interpreter。类中包含两个成员函数：

```
public:
    void translate(char *);
    void interpret(EmpList *emp);
```

图 4 interpreter 类成员函数

translate 函数功能：将输入的执行语句从字符串翻译成各类操作的状态参数。

Interpret 函数功能：根据当前状态参数，执行需求操作。

状态参数被定义在 interpreter 类的 private 中：

```
private:
    ATT attribute;
    OP operation;
    char* constrain;
```

图 5 interpreter 类状态参数定义

```
enum ATT{NAME, MY_IDNO, SEX, CELL, WAGE, TYPE};
enum OP{B, E, S, BE, SE, NE};//B-->, E--=, S--<, BE-->=, SE--<=, NE--!=
```

图 6 interpreter 类状态参数类型定义

参数的类型 ATT 代表操作执行的对象，类型 OP 代表操作的类别。

2.translate 函数实现

统一输入的指令的输入格式为“对象 操作 约束条件”。字符串传递进 translate 函数后，根据字符串内的空格符进行分段，将三段内容分别存入函数内定义变量 char str[300], char op[20] 和类内变量*constrain 中。接着执行对状态参数的翻译：

（1）用 strcmp 比较数组 str 内的内容根据比较结果给 interpreter.attribute 赋不同的值

“姓名”——NAME

“工号”——MY_IDNO

“手机”——CELL

“性别”——SEX

“工资”——WAGE

“人员类别”——TYPE

（2）用 strcmp 比较数组 op 内的内容根据比较结果给 interpreter.operation 赋不同的值。有一些无实际意义的查询语句不提供实现，比如“手机 >= 123”、“性别 < 男”、“姓名 != 张三”之类。

“>”——B

“<”——S

“=”——E

“>=”——BE

“<=”——SE

“!=”——NE

3.interpret 函数实现

首先对于*constrain 内保存的约束条件进行判定，若条件为数字，则将条件转变为 long long 类型的数字 num。

接着针对状态参数的值对不同的属性执行相应的操作，数据集由 EmpList *el 传递进来。函数实现的方法是，对 EmpNode 结构中的 nshow 指针和 lshow 指针进行操作。

定义 EmpNode *present 指向当前判断的指针，对 present->data 内的数据进行判断，若不符合条件将 present 的 lshow 指针指向的上个符合条件的节点的 nshow 指向 present 的下个节点。同时将下个节点的 lshow 指向 present 的 lshow 节点。搜索完成后，达到如下效果：EmpNode 结构中所有的 lall 和 nall 指针连接所有数据内容，lshow 和 nshow 指针连接所有符合操作需要输出的内容。

（四）界面设计与交互功能实现

界面框架在 DEV-C++内置的文本编辑器例程的基础上改造。

先介绍三个区域中比较简单的搜索区和统计区。搜索区的“当前结果中搜索”将在当前的搜索结果中继续搜索。实际上是一系列的逻辑与操作。

当一次搜索行为结束，需要返回以查看全部人员信息，或进行下一次搜索行为时，可点击“重置为所有”。

统计区的数据由遍历 EmpList 链表而计算得到。下方“更新统计信息”用途是，当载入二进制文件后，需要点击一下这个按钮，才会正常显示统计数据。

下面详细介绍员工信息管理区。

此区域的主体是 15 行列表项，每个列表项由一个选择按钮和 7 个单行文本框构成。这 7 个单行文本框由类 Row 管理。

Row 成员：

7 个 HWND 用于控制 7 个单行文本框。

Row 方法（一些简单的 Get 封装将略去）：

1. Display()。此函数接收一个 Employee*，将其信息翻译成字符串，展示到 7 个文本框中。

2. ShapeEmp()。此函数接收一个 Employee*。可看作 Display()的逆。

3. Empty()。此函数将 7 个文本框中内容清空。

4. Get48B()。此函数接收一个 char*，然后将 7 个文本框中的内容翻译成表示员工信息的 48B，存入字符串中。

另外，下方有 7 个按钮。下面解释每个按钮。

1. 上页与下页。翻页功能的实现利用了一个全局的 EmpNode*（名为 displayer）。这个指针指向的第一个点的数据被 Display()到第一行，它的下一个则进入第二行，依此类推，最多 15 个。点击上页或下页时，将这个 displayer 向前或向后移动 14 个节点的位置（有一条信息的重复，以示衔接），再重新显示即可。

2. 全选此页。对于每页的 15 条记录，是否选中是由一个全局数组 chosen[15]记录的。点击这个“全选此页”按钮，则可将这个数组元素全置 1；若已经都是 1，则全置 0。

3. 删除选中。对于 chosen[15]中为 1 的每个元素的对应 Row，调用 Empty()，同时调用 EmpList 的 Delete()，使他们从实际数据中被删去。

4. 确认更新。在对软件进行修改、插入后，点击这个按钮，系统将读取文本框中的内容，把当前显示的 15 条记录在 EmpList 中重新生成一遍，相当于数据的更新。如果发现有新的员工（原链表已经走到尾巴，但下一个文本框中还有内容），会先 new 一个 EmpNode，然后调用 EmpList 的 Insert。更新之后，这 15 行将从头（head 的 nshow 的指向）开始展示。

5. 奖、惩。为选中的员工调用 Reward() / Punish()虚函数，实现奖金控制。当然，奖金值也可以手工在文本框中输入。

二进制文件的存取点击 File 菜单项即可操作，略去。

五、组员分工

姓名	负责内容	具体代码
胡一夫	所有与界面交互相关的内容	main.h、main.cpp、empView.h、empView.cpp
岐舒骏	所有与搜索功能相关的内容	emp.h 与 emp.cpp 中涉及 interpreter 类的全部设计与实现，以及 EmpList 类中 Show() 和 SelectShow()两个方法
吴昊	与员工有关的所有类的设计	emp.h 中除去 interpreter 类的所有设计。
赵优	与员工有关的所有类的实现	emp.cpp 中的所有方法，但除去 interpreter 的方法和 EmpList 的 Show()、SelectShow()