**计 算 机 与 人 工 智 能 学 院**

**《数据结构》综合设计报告**

**设计名称：** **计划安排提醒系统**

**小组成员：**

**专业班级：**

**系（院）：**

**设计时间：**

**评价指标** **成绩:**

**1. 报告的规范性** **【规范】【较规范】【一般】【差】**

**2. 程序、算法的规范性 【规范】【较规范】【一般】【差】**

**3. 写作思路的清楚性 【清楚】【较清楚】【一般】【差】**

**4. 问题描述的清楚性 【清楚】【较清楚】【一般】【差】**

**5. 数据结构设计合理性 【合理】【较合理】【一般】【差】**

**6. 算法设计的合理性 【合理】【较合理】【一般】【差】**

**7. 完成情况** **【好】 【较好】 【一般】【差】**

**8. 创新性（或新意） 【有】 【无】**

**目录**

[**1**](#PageMark3)[**设计目的**](#PageMark3)

**1**

[**2**](#PageMark3)[**任务与要求**](#PageMark3) **1**

[**3**](#PageMark3)[**计划安排提醒系统**](#PageMark3) **1**

3.1 [系统（问题）描述](#PageMark3) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

3.2 [任务分配](#PageMark4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

3.3 [系统需求](#PageMark4) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

3.4 [设计思想](#PageMark5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

[3.4.1](#PageMark5) [数据结构的选择和设计](#PageMark5) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

[3.4.2](#PageMark7) [算法设计](#PageMark7) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3.5 [实现细节说明](#PageMark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

[3.5.1](#PageMark9) [数据的读取](#PageMark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

[3.5.2](#PageMark11) [界面的排版](#PageMark11) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

[3.5.3](#PageMark13) [数据的处理及结果反馈](#PageMark13) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

3.6 [测试数据及测试结果](#PageMark14) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

[3.6.1](#PageMark14) [当天提醒](#PageMark14) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

[3.6.2](#PageMark16) [提前三天提醒](#PageMark16) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

[**4**](#PageMark17)[**总结**](#PageMark17) **15**

4.1 [系统特点及创新](#PageMark17) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

4.2 [遇到的问题及解决思路](#PageMark17) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

I

**《数据结构》课程综合设计**

**1 设计目的**

在计算机科学中，数据结构是一般程序设计的基础。通过综合设计，使学生学会分析研究数据结构的特征，以便为应用涉及的数据选择适当的逻辑结构、存储结构及相应的算法，掌握算法的时间复杂度分析技术。另一方面，综合设计也是复杂程序设计的训练过程，要求学生编写的程序结构符合软件工程规范，培养他们的数据抽象能力、建模能力和算法设计能力，提高复杂问题的解决能力，为后续课程的学习和应用奠定基础。

**2 任务与要求**

要求学生以 5 人一组，自由结合，从给定的综合设计题目中进行选择。本次设计题目是设计内容不固定的题目，这就要求学生自己先确定本组设计题目，然后确定具体内容和设计目标，从而为数据结构设计、数据建模和算法设计奠定基础。提交的资料包含综合设计纸质版、电子版及源程序电子版。

**3 计划安排提醒系统**

**3.1 系统（问题）描述**

在现实生活中，大家都有许多的事情需要处理，并且由于有时候事务繁多，一个计划安排提醒的功能就会让人们能够及时的完成待办的事情。所以我们制作了一个“计划安排提醒系统”。他的主要功能是可以让用户自己添加相关的代办，并选择时间，事件。在这个事件要办理的当天以及前三天均会提醒用户，这很好的避免了用户忘记时间而没有完成相关的日程。

1

**3.2 任务分配**

我们小组四个人，每个人都在这个系统制作中是不可或缺的一部分。

1.小组长，主要负责事务分配，链表的构建以及相应添加，删除功能等函数的实现，以及文件读写和解决在制作过程中产生的一些 bug。

2.主要负责 EasyX 可视化部分的设计以及与链表功能的对接。并且实现了与现实时间比较的函数，并且在改进代码的功能过程中贡献了很大的力量。

3.主要负责链表数据的处理，比如时间的排序，以及数据的修改等函数的编写，也写出了计算页数的函数，给 EasyX 的页数计算提供了一定的基础。

4.主要负责日程提醒界面的设计以及修改，分别对三天前提醒以及当日提醒的功能编写了相关的函数。

对于课程设计文档的编写，是由我们四个人分工进行的，每个人编写自己熟悉

的部分，并且作业量都差不多。

在任务分配的过程中，我们主要考虑每个人拿手的技能，以及任务分配的平均性，来合理的分配任务，以此来保证每个人都能够参与到课程设计中来, 并且在课程设计过程中学到新的知识，巩固旧的知识。

**3.3 系统需求**

1. 具备添加日程安排的功能，其中需要添加的内容有日程的时间，以及日程的内容。

2. 具备删除日程安排的功能，其中删除功能具体到程序中应该体现为两个方面，分别是完成了日程，以及针对错误日程或者想要取消日程的删除。

2

3. 具备修改日程安排的功能，因为数据内容的特殊性，要求能够结合实际合理地实现该功能。

4. 拥有提醒功能，能对用户尚未完成的日程进行判断，若当日有待办事项，应该提醒用户；若有临近日程，例如三日后的日程，也应该提醒用户；若今日无待办事项，也应当提醒用户，以便用户清楚自己的日程。

5. 能够以合理的形式显示安排中的日程，使用户对于自己的行程能够一目了然。

**3.4 设计思想**

本系统以用户体验为核心，将用户的使用感受放在第一位，选用更为用户接受的图形化界面，将一切复杂的数据以合理美观的形式显示在用户眼前，同时根据日程数据特殊的形式以及需要频繁地查改以及数据数目的不确定性等的特点，本系统选用链表结构实现对数据的存储和操作；在开发过程中，我们针对本系统体量较小的特点，采用合作开发模式，将软件需求模块化后分别进行代码的编写，最后将数据操作以及 UI 模块进行整合，形成完整的程序。

**3.4.1 数据结构的选择和设计**

数据结构：链表，链表中的数据采用的是结构体类型的数据

*#include<stdio.h>*

*//*定义链表所存数据**typedef struct** Content **{**

**char** thing**[100];**

3

**char** time**[15]={0}; bool** preReming**;**

**}**content**;**

*//*定义链表节点

**typedef struct** ThingNode **{**

content data**;**

**struct** ThingNode**\*** next**;**

**int** length**; }**List**;**

*//*创建链表

List**\*** creat\_list**() {**

List**\*** node **= (**List**\*)**malloc**(sizeof(**List**));** node**->**next **=** NULL**;**

node**->**length **= 0;**

**return** node**;**

**}**

采用链表来储存日程的相关数据，主要是因为各个日程之间没有什么相关性，单个独立，并且采用链式存储查找与删除都十分方便。采用链表可以更加高效的储存数据

4

**3.4.2 算法设计**

1. 冒泡排序算法：是一种稳定排序算法，通过冒泡排序算法，稍微根据数据的类型进行了一定的修改，将日程的时间作为关键字，对日程进行排序。使日程更加的有序，便于查看

**void** ThingSort\_ByTime**(**List**\*** head**) {**

*//*按照日期从早到晚排序

*//*采用冒泡法进行排序

List**\*** front**, \*** middle **, \*** rear**;**

**for (int** i **= 0;** i **<** head**->**length **- 1;** i**++) {**

front **=** head**,** middle **=** head**->**next**,** rear **=** middle**->**next**;**

**for (int** j **= 0;** j **<** head**->**length **- 1 -** i**;** j**++) {**

**if (**strcmp**(**middle**->**data**.**time**,** rear**->**data**.**time**) > 0) {**

middle**->**next **=** rear**->**next**;** rear**->**next **=** middle**;** front**->**next **=** rear**; }** front **=** front**->**next**;** middle **=** front**->**next**;**

rear **=** middle**->**next**; }**

**}**

**}**

2. 链表操作的算法：比如修改日程根据链表数据的修改进行，由于这个链表数据内容较多，所以采用逐个输入的办法，通过 EasyX 来进行可视化，通过按钮等函数来进行可视化交互。还有链表数据的插入，删除等。

5

3.UI 算法：在修改日程，以及日程已完成之后可以点击日程行后边的按钮，来完成上述操作。通过 EasyX 函数库，根据鼠标位置同时结合位置来判断数据在链表中的位置，来对该数据进行操作。其他的 UI 操作还有“按钮”，“弹出界面”等。

**void** things\_finish**(**ExMessage msg**) {**

**int** i **= 35,** j **= 0;**

**for (**j**;** j **< 15;** j**++) {**

**if (**msg**.**x **>= 1030 &&** msg**.**y **>= 140 +** i **\*** j **&&** msg**.**x **<= 1080 &&** msg**.**y **<= 170 +** i **\*** j**)**

clearroundrect**(1032, 140 +** i **\*** j**, 1080, 171 +** i **\*** j**, 0, 0);**

button\_delete**(1032, 142 +** i **\*** j**, 1078, 169 +** i **\*** j**,** " 已完成"**,** YELLOW**,** skyblue**);**

**if (**msg**.**message **==** WM\_LBUTTONDOWN**) {** delete\_list**(**ThingList**,** j**+1,** page\_cnt**);** show\_things**();}**

**}**

**else {**

button\_delete**(1032, 140 +** i **\*** j**, 1080, 171 +** i **\*** j**,** ""**,** YELLOW**,** gray**);}**

**}**

write\_file**(**ThingList**,** "thing.txt"**);**

**}**

4. 翻页算法：根据链表长度将数据分页，更加清楚的展示日程。根据数据选择的页

面刷新显示相应的页面。

5.time 函数算法：通过对时间的英文首字母比对，将时间存放在 Temp 数组中，并最后将 Temp 数组转化为字符串。

6

**3.5 实现细节说明**

系统的关键算法可分为三个部分：数据的读取、界面的排版、数据的处理及结

果反馈

**3.5.1 数据的读取**

运用链表，便于增删修改。其中，用头插法实现数据的增加，并用冒泡排序来

确定新增数据的插入位置。

1. 头插法：

*//*添加日程 *(*头插法*)*

**void** add\_list**(**List**\*&** head**,** Content data**) {**

List**\*** node **=** creat\_list**();** node**->**data **=** data**;** node**->**next **=** head**->**next**;** head**->**next **=** node**;** head**->**length**++;**

**}**

2. 冒泡排序

**void** ThingSort\_ByTime**(**List**\*** head**) {**

*//*按照日期从早到晚排序

*//*采用冒泡法进行排序

List**\*** front**, \*** middle **, \*** rear**;**

7

**for (int** i **= 0;** i **<** head**->**length **- 1;** i**++) {**

front **=** head**,** middle **=** head**->**next**,** rear **=** middle**->**next**; for (int** j **= 0;** j **<** head**->**length **- 1 -** i**;** j**++) {**

**if (**strcmp**(**middle**->**data**.**time**,** rear**->**data**.**time**) > 0) {**

middle**->**next **=** rear**->**next**;** rear**->**next **=** middle**;** front**->**next **=** rear**;**

**}**

front **=** front**->**next**;** middle **=** front**->**next**;** rear **=** middle**->**next**;**

**}**

**}**

**}**

排序后的日程如下图所示：

8



**3.5.2 界面的排版**

应用 easyx 来进行界面设计的大部分内容，例如界面的长宽、背景颜色、字体大小颜色、按键的大小和颜色等。（以下以系统首页为例）

1. 界面的长宽：

initgraph**(500, 500);**

2. 背景颜色：

setbkcolor**(**RGB**(225, 245, 245));**

3. 字体大小颜色：

settextcolor**(**RGB**(225, 0, 245));**

settextstyle**(24, 0,** " 宋体"**);**

9

outtextxy**(175, 150,** " 今日无待办事项"**);**

4. 按键大小颜色：

*//*按钮函数

**void** button**(int** left**, int** top**, int** right**, int** bottom**,**

**const char\*** flag**,** COLORREF word**,** COLORREF back**)**

**{**

**int** x**,** y**;** settextcolor**(**word**);** setfillcolor**(**back**);** setbkmode**(**TRANSPARENT**);**

settextstyle**(24, 0,** " 宋体"**);** solidroundrect**(**left**,** top**,** right**,** bottom**, 20, 20);** x **=** left **+ (**right **-** left **-** textwidth**(**flag**)) / 2;** y **=** top **+ (**bottom **-** top **-** textheight**(**flag**)) / 2;** outtextxy**(**x**,** y**,** flag**);**

**}**

5. 鼠标对界面的控制:

ExMessage msg**;** peekmessage**(&**msg**,** EM\_MOUSE**);**

**if (**msg**.**x **>= 210 &&** msg**.**y **>= 300 &&** msg**.**x **<= 290 &&** msg**.**y **<= 350) {**

clearroundrect**(210, 300, 290, 350, 20, 20);**

button**(210, 300, 290, 350,** " 好的"**,** dark\_orign**,** RED**);**

10



**if (**msg**.**message **==** WM\_LBUTTONDOWN**) {**

**break;**

**}**

**}**

**else {**

button**(210, 300, 290, 350,** " 好的"**,** dark\_orign**,** skyblue**);**

**}**

日程提醒界面图示：

**3.5.3 数据的处理及结果反馈**

1. 本机时间的读取:

具体使用 time 函数，获取本机时间并转化为标准格式。因程序进行运算时使用的是字符串，所以再通过 ReadNowTime 函数中部分代码实现了从标准格式到字

11



符串的转换。

本机时间的读取和转换:

*#include<windows.h>* **time\_t** rawtime**; struct** tm**\*** timeinfo**;** time**(&**rawtime**);**

timeinfo **=** localtime**(&**rawtime**);**

2. 提醒时间的判断:

系统共有两个提醒时间，一为当天，二为提前三天，其中判断当天与否只需比较列表中的字符串与当前时间的字符串中前八位字符是否相同即可；提前三天的需要进行字符串的拆分比较，期间需要考虑年份月份等问题。

**3.6 测试数据及测试结果**

**3.6.1 当天提醒**

输入样例：

12



输出样例：

13



**3.6.2 提前三天提醒**

输入样例：

输出样例：

14



**4 总结**

**4.1 系统特点及创新**

本系统采用 EasyX 图形库，来达到页面可视化的效果。通过使用 EasyX 所包含的头文件 graphics.h 来绘制页面窗口及按钮，另外使用 time.h 头文件，读取计算机现在的时间，从而与用户输入的时间相比较来判断是否达到用户设置的提醒时间，此外，用户还可以选择是否需要提前三天进行提醒。用户可以首先输入乱序的所有需要提醒的事项，然后可以进行一键根据输入的时间自动进行对事件依据时间进行排序。

**4.2 遇到的问题及解决思路**

遇到的困难 1: 如何对用户输入的所有事件根据其事件进行排序，因为在创建链表用以存储用户输入数据时，为创建的单链表，解决方案初始化了三个指针，用

15

以指示待删除结点的头结点、待删除结点和待删除结点的下一节点，在删除结点后，第一指针直接指向第三指针。

遇到的困难 2：时间表示的选择问题，在程序内的时间可以是预置时间也可以和本计算机的时间一致，在时间的选择上我们选择了使用 time.h 头文件使程序内时间为本计算机时间。

遇到的困难 3：在事件的选择上，我们本想使用音乐播放函数，使事件达到用户指定时间时播放音乐以提醒用户，但由于该程序并不是一直运行，只在用户打开本程序时运行，所以我们将提醒方式改为在某一天需要所的事情，而并非是在该做事情的时间提醒。

遇到的困难 4：多人项目合作难以及多人合作变量问题。由于小组每个人的编程习惯并不相同，因此在各个任务程序的协调性上存在难题，最终经过不断的交流和沟通，对各部分代码的讨论和修改我们最终解决了这些困难。

遇到的困难 5：代码注释，为了使设计不同功能的同学间可以相互调用功能函数，代码注释是必要的。解决的方法是，每位同学在编写代码时尽量做到每个功能所用的方法和每部分代码作用给予注释，同时，必要的注释也为我们减少了不必要的交流沟通，从而提高了编程和调试的效率。

16