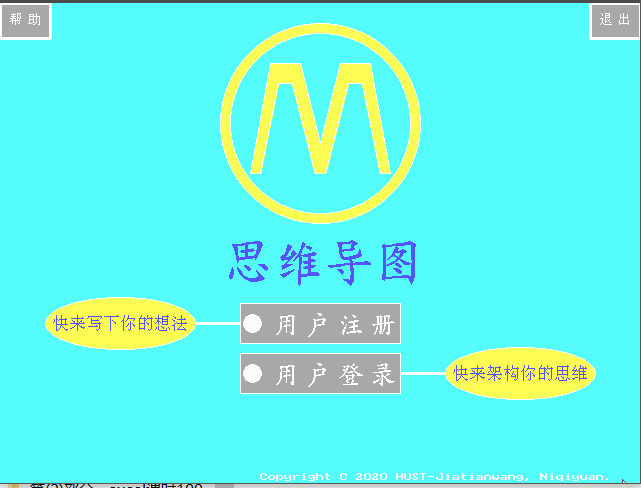
**思维导图**

**C语言课程设计报告**



**专业班级：自动化1903班**

**小组成员：贾田旺 U201914626**

**倪启源 U201914631**

**指导老师：周纯杰 何顶新 彭 刚**

**周凯波 桑 农 左峥嵘**

**高常鑫 汪国友 陈 忠**

目录

[目录 1](#_Toc54854540)

[1．引言 1](#_Toc54854541)

[1.1 编写目的 1](#_Toc54854542)

[1.2 背景 1](#_Toc54854543)

[1.3 目标 2](#_Toc54854544)

[1.4 客户特点 2](#_Toc54854545)

[1.5 运行环境与配置 3](#_Toc54854546)

[1.5.1 硬件接口 3](#_Toc54854547)

[1.5.2 软件接口 3](#_Toc54854548)

[1.5.3 控制 3](#_Toc54854549)

[1.6 运行规范 4](#_Toc54854550)

[1.7 编者的话 4](#_Toc54854551)

[2．软件需求分析 5](#_Toc54854552)

[2.1 软件需求概述 5](#_Toc54854553)

[2.2 软件各功能介绍 8](#_Toc54854554)

[2.2.1 登陆注册功能 8](#_Toc54854555)

[2.2.2 思维导图操作功能 8](#_Toc54854556)

[2.2.3 新建思维导图功能 9](#_Toc54854557)

[2.2.4 打开思维导图功能 9](#_Toc54854558)

[2.2.5 关闭及删除思维导图功能 9](#_Toc54854559)

[2.2.6 思维导图编辑功能 9](#_Toc54854560)

[2.2.7 帮助功能 10](#_Toc54854561)

[3．界面设计与界面逻辑（以下图案非最终效果） 10](#_Toc54854562)

[3.1 欢迎界面 11](#_Toc54854563)

[3.2 注册界面 12](#_Toc54854564)

[3.3 登录界面 14](#_Toc54854565)

[3.4 密码修改界面 16](#_Toc54854566)

[3.5 思维导图用户文件操作主界面 17](#_Toc54854567)

[3.6 新建界面 19](#_Toc54854568)

[3.6 打开界面 21](#_Toc54854569)

[3.7 帮助界面 22](#_Toc54854570)

[3.8 编辑界面 23](#_Toc54854571)

[5．模块设计 26](#_Toc54854572)

[5.1 登录模块 26](#_Toc54854573)

[5.2 注册模块 27](#_Toc54854574)

[5.3 密码修改模块 29](#_Toc54854575)

[5.4 操作模块 30](#_Toc54854576)

[5.5 新建模块 31](#_Toc54854577)

[5.6 打开模块 32](#_Toc54854578)

[5.7 帮助模块 33](#_Toc54854579)

[5.8 编辑模块 34](#_Toc54854580)

[5.8.1 编辑单向思维导图 34](#_Toc54854581)

[5.8.2 编辑树状图 35](#_Toc54854582)

[5.8.3 编辑鱼骨图 36](#_Toc54854583)

[5.8.4 编辑时间轴 36](#_Toc54854584)

[6．数据结构和数据流 37](#_Toc54854585)

[7．感想与体会 40](#_Toc54854586)

[组员贾田旺的感想和反思： 40](#_Toc54854587)

[组员倪启源的感想和反思： 42](#_Toc54854588)

[8．出现的问题及解决办法 44](#_Toc54854589)

[8.1 用户的文件管理问题 44](#_Toc54854590)

[8.2 思维导图核心功能的实现问题 44](#_Toc54854591)

[9．源代码 47](#_Toc54854592)

[9.1 welcome.c 47](#_Toc54854593)

[9.2 Signup.c 54](#_Toc54854594)

[9.3 ResetKey.c 62](#_Toc54854595)

[9.4 Log.c 70](#_Toc54854596)

[9.5 Help.c 78](#_Toc54854597)

[9.6 EnterT.c 88](#_Toc54854598)

[9.7 Operate.c 104](#_Toc54854599)

[9.8 OperateT.c 118](#_Toc54854600)

[9.9 Opena.c 127](#_Toc54854601)

[9.10 newa.c 132](#_Toc54854602)

[9.11 Edit.c 141](#_Toc54854603)

[9.12 Boxa.c 151](#_Toc54854604)

[9.13 Boxadl.c 156](#_Toc54854605)

[9.14 Edittr.c 162](#_Toc54854606)

[9.15 Boxtr.c 170](#_Toc54854607)

[9.16 Boxtrdl.c 175](#_Toc54854608)

[9.17 Editfb.c 180](#_Toc54854609)

[9.18 Boxfb.c 185](#_Toc54854610)

[9.19 Edidate.c 188](#_Toc54854611)

[9.20 Boxdate.c 196](#_Toc54854612)

[9.21 EditT.c 201](#_Toc54854613)

[9.22 main.c 208](#_Toc54854614)

[10．时间安排及分工 209](#_Toc54854615)

[10.1 组员分工 209](#_Toc54854616)

[10.2 组员文件统计与代码量 210](#_Toc54854617)

[10.3 时间安排 211](#_Toc54854618)

1．引言

1.1 编写目的

思维导图作用于生活和工作的各个方面，包括学习、写作、沟通、演讲、管理、会议等，运用思维导图带来的学习能力和清晰的思维方式会改善人的诸多行为表现：成倍提高学习速度，更快地学习新知识与复习整合旧知识；激发联想与创意，将各种零散的智慧、资源等融会贯通成为一个系统；形成系统的学习和思维的习惯，思维导图必将成为未来不可或缺的教学和学习工具。

我们希望本款思维导图软件能够帮助我们整理思维，纵情表达。

1.2 背景

随着科技的发展，如今这个时代是一个信息爆炸的时代，一天内增长的信息会是过去数年之多。随着我们获取信息的途径越来越多，获取的信息愈加增多，但是整理信息的工具却没有产生。在这样的大趋势之下，思维导图作为一种整理信息的工具，应时而生。

在运用思维导图之后，可以迅速整理我们获得的信息，搭建合适的结构，便于以后的浏览和复习。同时，思维导图能够帮助我们厘清我们所获得的信息的大致思路，使我们的思路更加清晰明了，从而加深我们对信息的认识与理解。

思维导图的作用还不止于此。在会议上，演讲中，无时无刻不存在思维导图的运用，向别人展示你的思维，向别人诉说你的想法，更是离不开思维导图。思维导图将各种零散的想法整合到一起，诉诸以整体架构，让别人能够在较短的时间之内掌握表述人的思想。如此看来，思维导图必将成为重要的教学和学习工具。

此外，在运用思维导图的过程中，我们对所获得的信息加以整理，使之形成模块化、结构化的知识。事实上，在无形中，我们也在进行一定的思维训练。在以后遇到新的知识后，我们将会主动在脑海里构建思维模型，对知识加以处理。因此在学习使用思维导图之后，会更好地锻炼思维，使我们的思维更加有序、高效。

1.3 目标

1. 本软件在能够满足基本思维导图编辑需求的基础上，尽力完善功能，通过不同的思维导图编辑模式，适应不同场景的应用，尽可能地简化操作，能够流畅运行。

2. 通过软件的开发过程，了解掌握软件开发的方法、技巧和基本流程，更深入的理解软件的实现逻辑。

3. 本软件对文件和内存的使用要求较高，且需要一定的算法，通过编写本软件加深算法理解能力与自学能力，同时尽可能优化对文件内存的使用方案。

4. 本软件面向年轻客户人群，软件设计独立，拥有友好的图形化界面，友好的提示与帮助和紧跟网络潮流的文案描述，软件设计尽量美观，方便。

5. 通过分组编写软件加强分工协作能力与沟通能力。

1.4 客户特点

1. 本软件面向的人群为新时代处于学习、工作阶段，对信息的获取和整理有较高要求，且愿意接受新事物并学习去运用的人。

2. 操作及维护人员：具有良好的C语言编程技术，能够充分了解软件模块，并且熟练掌握软件的数据结构。

3. 软件与其使用频率：大面积推广后，在上述面向的人群中有较大影响力，并且能够吸引客户天天使用，形成整理及锻炼思维的习惯。

1.5 运行环境与配置

1.5.1 硬件接口

处理器：Intel Pentium 166 MX 或以上。

硬盘：空间500MB以上。

屏幕适配器：VGA接口。

系统运行内存：要求32MB以上。

1.5.2 软件接口

开发软件工具：Borland C 3.1。

文字编辑工具：Notepad++、Visual Studio Code。

操作系统：DOS

1.5.3 控制

该系统通过鼠标与键盘直接进行控制。用户将鼠标移至需要操作的功能区进行点击，同时通过键盘来完成登陆、注册的输入功能。操作完毕点击相应位置退出系统。通过中断技术来获取鼠标的位置与键盘的输入功能。

1.6 运行规范

1. 命名规范：

变量命名，涉及用户以及思维导图信息的，应该尽量用英文表达其准确定义。其他类型变量名应给出详细注释以说明其主要功能。

函数命名应该用英文表达其确定含义。

文件命名用大，小写，并且表达出该文件所包含函数的主要功能。

2. 注释：

函数功能都要在函数原型后注明。

部分令测试者比较难以理解的算法和流程应该给出相应的注释。

在每个文件头部要给出本文件的功能注释。

1.7 编者的话

选择思维导图软件任务是我们经过认真考虑的结果，我们认为思先于行，梳理好自己的思维后，完成任务有事半功倍的效果。

通过对市面上热门的思维导图软件的调研分析，尝试使用，并翻阅一些资料后，我们经过多次讨论和审慎的决策，编写出了这一份软件需求分析和功能设计报告。

该项报告从多个角度，对整个思维导图软件的用户需求和功能分析作了详细的阐述。具体包括如下几个方面：编写背景、目标、受众等方面的概述，软件需求分析（大致介绍了软件设计的一些思路），软件各功能介绍，界面的设计，目前已实现功能的操作介绍，软件分模块的功能设计方案以及系统的详细设计等。

与此同时，本报告明确了两位开发者的任务分工。

本报告的预期受众为对思维训练有一定需求的人以及对大量信息的快速处理有需要的人（如会计、教师、学生等），此外本报告也面向软件开发从业者等人。

**参考资料：**

1. 王士元. C高级实用程序设计. 北京： 清华大学出版社. 1996年

2. 周纯杰，何顶新等. 程序设计与应用（用C/C++编程）. 北京： 机械工业出版社. 2008年

3. [美] Prata. C Primer Plus（第六版）北京：人民邮电出版社. 2016年

4. 严蔚敏，吴伟民编著. 数据结构（C语言版）北京：清华大学出版社. 2018年

**参考软件：**

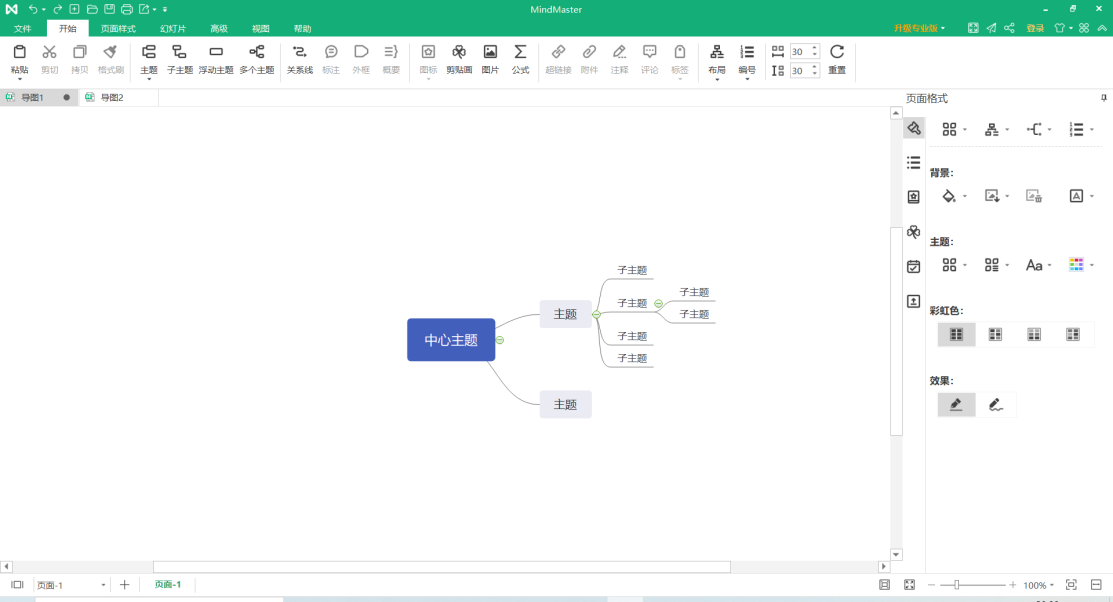
Xmind思维导图

MindMaster 思维导图软件

2．软件需求分析

2.1 软件需求概述

本软件旨在为人们提供一种方便快捷，容易操作的思维工具。为达到此目的，我们参考了多种思维导图软件。如MindMaster等。在参考Mindmaster软件的过程中，我们一致认为其思维导图编辑界面美观，大方，简洁，值得借鉴。因此，我们在设计思维导图编辑页面的时候借鉴了MindMaster的界面编辑思路：在界面的右侧放置思维导图编辑的操作栏，而在界面的上方显示思维导图的名称。MindMaster界面如图所示：



图表 1 MindMaster参考界面

此外，我们综合研究了各大软件所提供的思维导图模板之后，认为单向导图、树状图、时间轴、鱼骨图这四种图在使用上较为方便，且受众面较广。因此，我们选用了这四种思维导图模板提供给使用者使用。

思维导图最重要的功能就是对导图的框的编辑，我们通过对各种思维导图软件的使用，发现思维导图框的编辑操作至少需要包含对框内文字的编辑、对框的形状颜色的编辑、对框的增，删等方面的内容。因此，我们当前确定了对思维导图编辑的几种操作（增、删、文字编辑等），并正在设计算法思路来完成相应的设想。MindMaster的一些功能如图所示：



图表 2 MindMaster功能

除了思维导图核心功能的实现以外，为提供给用户良好的使用体验并方便用户对自己的文件进行管理，我们设计了登录、注册、找回密码，新增、打开、删除、关闭文件等功能。

我们还在许多方面对软件的设计进行了借鉴，并查阅大量资料，了解当下人们的需求，结合这些资料，我们着力于满足人们的需求，努力做出一款受到人们认可的软件。

以下，我们将介绍各功能的设计方案。

2.2 软件各功能介绍

2.2.1 登陆注册功能

用户打开软件后首先进入的是欢迎界面，用户可以选择注册或登录。若是新用户，需点击“用户注册”按钮，进入注册界面，需依次输入用户名、身份证号、密码、确认密码进行注册，点击“注册”完成注册，其中若出现格式不正确，两次输入密码不相符等错误，界面会提示错误信息。完成注册后，将跳回欢迎主界面。用户可以点击“用户登录”按钮进入登陆界面，通过输入用户名及密码，点击“登录”按钮完成登录，其中若出现用户名不存在、密码输入错误等错误，界面会提示错误信息。若忘记密码，用户可以点击“忘记密码”按钮进入重置密码界面，通过输入用户名、身份证号、密码、确认密码信息，点击“确定”按钮进行密码重置，其中若出现用户名不存在，两次输入密码不相符等错误，界面会提示错误信息。完成重置后，将跳回登陆界面。用户在完成注册后会生成一个以用户名为名字的文件用以存储该用户的导图信息。

在本界面，用户可以点击“帮助”查看软件帮助信息。

2.2.2 思维导图操作功能

用户登录后进入思维导图操作主界面。

首先用户可以选择新建或打开一个思维导图，还可以对打开的导图进行关闭或删除。

在新建或打开一个导图后将跳回操作主页面，并显示思维导图的信息。

2.2.3 新建思维导图功能

用户在操作思维导图界面点击“新建”进入思维导图新建界面。通过输入思维导图的名称并选择思维导图的类型完成新建，每个不同的思维导图类型对应不同的记录与显示方式。其中如果出现未对思维导图种类进行选择、名称不符合要求等错误，界面会提示错误信息。完成新建后，将跳回操作界面。并显示所编辑导图的名称等信息。用户每新建一个导图，会将导图种类，名称等信息存入以用户名命名的文件，并新建一个以导图名命名的文件以存储导图信息。

2.2.4 打开思维导图功能

用户在操作思维导图界面点击“打开”进入思维导图打开页面。通过输入思维导图的名称完成打开。其中如果出现导图不存在错误，界面会提示错误信息。完成打开后，将跳回操作界面。并显示所编辑导图的名称等信息。

2.2.5 关闭及删除思维导图功能

用户在操作思维导图界面点击“关闭”将关闭已打开的思维导图，操作界面回到未打开思维导图状态。

用户在操作思维导图界面点击“删除”将关闭已打开的思维导图，并删除以导图名命名的文件，在以用户名命名的文件中删除该导图名称、种类等信息。之后，操作界面回到未打开思维导图状态。

2.2.6 思维导图编辑功能

用户在操作思维导图界面点击编辑按钮可以进入编辑界面。选中思维导图框后，首先进入编辑文本模式，可以对思维导图内容进行编辑，点击“回车”后退出文本编辑模式，之后可以新增框的子代，或删除选中框，导图的界面显示会随之调整。

2.2.7 帮助功能

用户在欢迎界面点击帮助按钮可以进入帮助界面，查阅软件使用帮助信息。

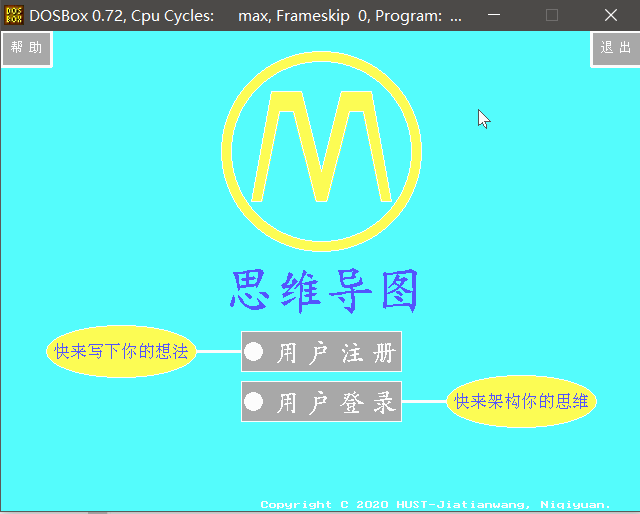
3．界面设计与界面逻辑（以下图案非最终效果）

为了方便用户操作，同时让用户有一个良好的使用体验，我们经过讨论，决定以不牺牲操作简便性为前提，将界面做到简洁、美观。

从总体上来讲，整个课程设计的每一个页面我们都采用了大致相同的色调和标签风格等进行设计，每一个页面看似风格一致而又不乏变化，带给用户一种新奇的使用体验，但又不至于因变化过多而使得使用者无所适从。在设计每一个页面的时候，我们都会反复调试，对参数进行适当的调整，使每一个按键，图案，输入框的位置尽可能符合大多数人的使用习惯。

以下，我们将对每一个界面，分别加以介绍：

3.1 欢迎界面



图表 3 欢迎界面

欢迎界面：欢迎界面由思维导图logo及名称，用户注册按钮、用户登录按钮、退出按钮及帮助按钮部分组成。

欢迎界面操作：

（1）点击【用户注册】进入注册界面

鼠标悬停在【用户注册】按钮上该按钮左侧蓝色的“快来写下你的想法”变为粉红色的“点击即可加入我们”，周围的白框变为粉红色，用户注册四字与其左侧的圆圈也变为粉红色。

（2）点击【用户登录】进入登录界面

鼠标悬停在【用户登录】按钮上该按钮右侧蓝色的“快来架构你的思维”变为粉红色的“登录即可纵情表达”，周围的白框变为粉红色，用户登录四字与其左侧的圆圈也变为粉红色。

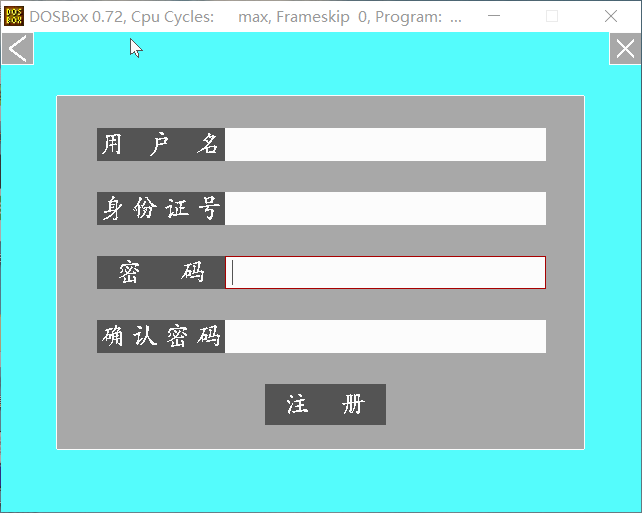
（3）点击【退出】退出程序

鼠标悬停在【退出】按钮上该按钮周围的白框变为红框。

（4）点击【帮助】按钮进入到帮助界面

在此界面可以查看软件帮助信息。

3.2 注册界面



图表 4 注册界面

注册界面：注册界面由用户名、身份证号、密码、提示密码四个部分的提示框及各自的输入框，注册按钮，返回按钮（<），退出按钮（x）组成。

注册界面操作：

（1）进入注册界面，输入用户名、身份证号、密码、确认密码后点击【注册】可能出现如下状况：

A.信息输入完全

a.输入的用户名未注册过，身份证号为18位且格式正确，两次输入的密码一致且长度大于等于8位：将用户信息存储到Userinfo.dat文件夹中，同时新建一个以用户名命名的，存放用户思维导图的文件夹于daotu文件夹中；同时返回到欢迎界面。

b.输入的用户名注册过：在用户名输入框的右下方出现“用户名已存在”的提示信息。

c.用户名未注册过，身份证号不符合格式：在身份证号输入框的右下方出现“身份证号格式错误”的提示信息。

d.用户名未注册过、身份证号符合格式，密码不符合要求：在密码输入框的右下方出现“密码长度至少为八位”的提示信息。

e.用户名、身份证号、密码均符合上述要求，确认密码与密码不一致：在确认密码输入框的右下方出现“两次输入密码不一致”的提示信息。

B.信息输入不完全

a.未输入用户名：在用户名输入框的右下方出现“未输入用户名”的提示信息。

b.输入了用户名，未输入身份证号：在身份证号输入框的右下方出现“未输入身份证号”的提示信息。

c.输入了用户名、身份证号，未输入密码：在密码输入框的右下方出现“未输入密码”的提示信息。

d.输入了用户名、身份证号、密码，未输入确认的密码：在确认密码输入框的右下方出现“未确认密码”的提示信息。

（2）点击【返回】（<）回到欢迎界面。

（3）点击【退出】（x）退出程序。

3.3 登录界面



图表 5 登陆界面

登录界面：登录界面由用户名提示框及输入框，密码提示框及输入框，登录按钮，忘记密码按钮等部分组成。

登录界面操作：

（1）进入登录界面，输入用户名密码后点击【登录】可能出现如下状况：

A.信息输入完全

a.用户名、密码输入正确：进入编辑界面。

b.输入了未注册过的用户名：在用户名输入框的右下方出现“用户名不存在”的提示信息。

c.输入了注册过的用户名，但密码错误：在密码输入框的右下方出现“密码错误”的提示信息。

B.信息输入不完全

a.未输入用户名：在用户名输入框的右下方出现“未输入用户名”的提示信息。

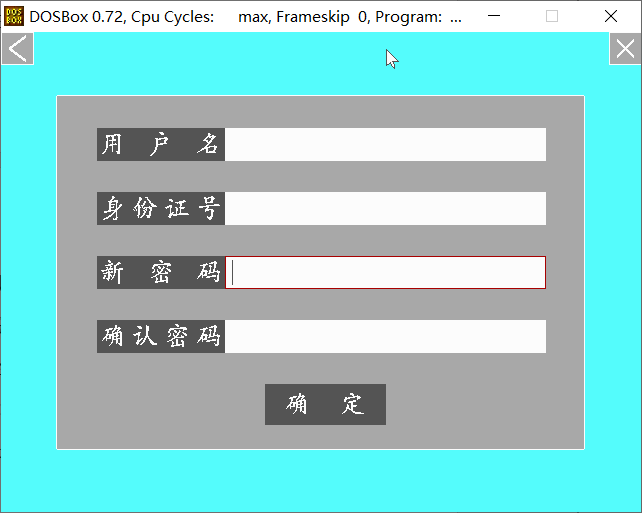
b.输入了用户名，未输入密码：在密码输入框的右下方出现“未输入密码”的提示信息。

（2）点击【忘记密码】，进入密码修改界面。

（3）点击【返回】（<）回到欢迎界面。

（4）点击【退出】（x）退出程序。

3.4 密码修改界面



图表 6 密码修改界面

密码修改界面：密码修改界面由用户名、身份证号、新密码、提示密码四个部分的提示框及各自的输入框，确定按钮，返回按钮（<），退出按钮（x）组成。

密码修改界面操作：

（1）进入密码修改界面，输入用户名、身份证、新密码、确认密码后点击【确定】可能出现如下状况：

A.信息输入完全

a.输入的用户名存在，身份证号为对应用户名的身份证，两次输入的密码一致且长度大于等于8位：在用户信息文件Userinfo.dat中修改用户的密码信息；返回至登录界面。

b.输入的用户名不存在：在用户名输入框的右下方出现“用户名不存在”的提示信息。

c.用户名存在，身份证号不是用户名对应的：在身份证号输入框的右下方出现“身份证号错误”的提示信息。

d.用户名、身份证号符合要求，密码长度小于八位：在密码输入框的右下方出现“密码长度至少为八位”的提示信息。

e.用户名、身份证号、密码均符合上述要求，确认密码与新密码不一致：在确认密码输入框的右下方出现“两次输入密码不一致”的提示信息。

B.信息输入不完全

a.未输入用户名：在用户名输入框的右下方出现“未输入用户名”的提示信息。b.输入了用户名，未输入身份证号：在身份证号输入框的右下方出现“未输入身份证号”的提示信息。

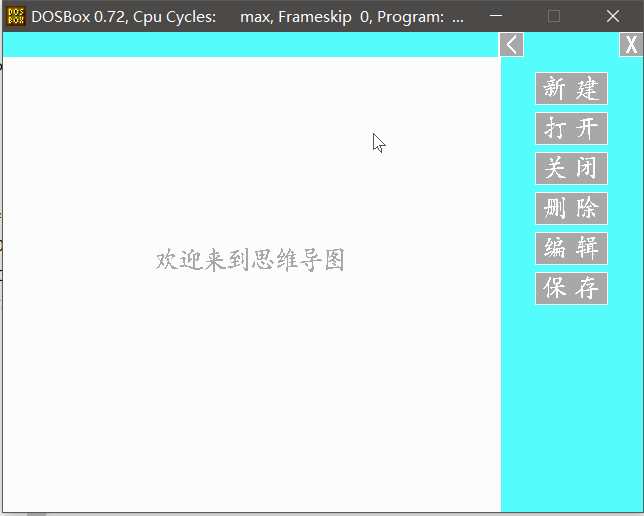
c.输入了用户名、身份证号，未输入新密码：在新密码输入框的右下方出现“未输入密码”的提示信息。

d.输入了用户名、身份证号、新密码，未输入确认的密码：在确认密码输入框的右下方出现“未确认密码”的提示信息。

（2）点击【返回】（<）回到欢迎界面。

（3）点击【退出】（x）退出程序。

3.5 思维导图用户文件操作主界面



图表 7 思维导图操作界面

操作界面：操作界面由“新建”“打开”“关闭”“删除”“编辑”“帮助”“返回”“退出”等按钮以及思维导图展示界面组成。

操作界面操作：

（1）点击【新建】进入新建界面。

（2）点击【打开】进入打开界面。

（3）点击【关闭】关闭当前打开的文件。

（4）点击【删除】删除当前打开的文件。

（5）点击【编辑】进入思维导图编辑界面。

若当前没有打开思维导图，则不进入思维导图操作界面

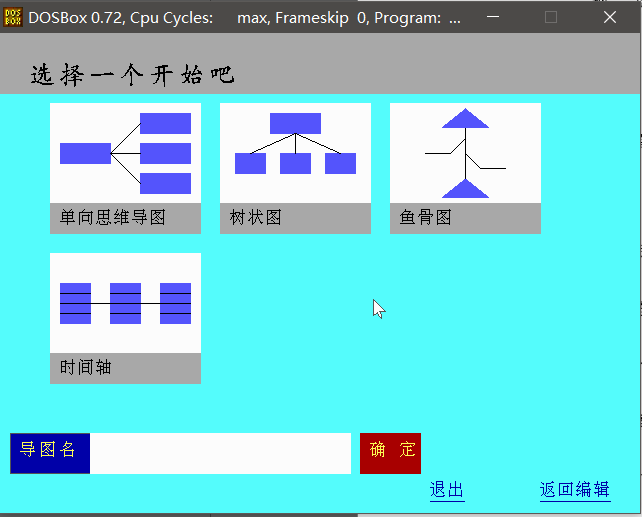
（6）点击【保存】保存所做更改。

（7）点击【返回】（<）回到欢迎界面。

（8）点击【退出】（x）退出程序。

（9）新建、打开等操作若成功，导图显示区域的上方将出现当前思维导图的名称，同时下方导图显示区域将呈现当前导图。

3.6 新建界面



图表 8 新建思维导图界面

新建页面：新建页面由“选择一个开始吧”标题、单向思维导图按钮、树状图选择按钮、导图名及输入框、确定按钮、退出按钮、返回编辑按钮组成。

新建页面操作：

（1）进入新建界面，选择思维导图，输入思维导图名称后点击【确定】可能出现如下状况：

A.信息输入完全

a.已选择某一思维导图（对应的思维导图名称会标亮），且输入了导图名称，导图名称长小于等于八字符：进入编辑界面，编辑界面上方出现新建的导图名称，在daotu文件夹中对应用户的文件夹下新建一个.dat文件存放新建的思维导图信息。

b.导图名称长大于八字符：在导图名输入框下出现“最多八字符”的提示信息。

B.信息输入不完全

a.未输入新建导图名：在导图名输入框下出现“未输入新建文件名”的提示信息。

b.输入了新建导图名，未选择导图种类（无导图被标亮）：在导图名输入框下出现“选个种类吧”的提示信息。

（2）点击【退出】退出程序。

（3）点击【返回编辑】回到编辑界面。

3.6 打开界面



图表 9 打开界面

打开界面：打开界面由输入框，确定打开按钮以及取消按钮组成。

打开页面操作：

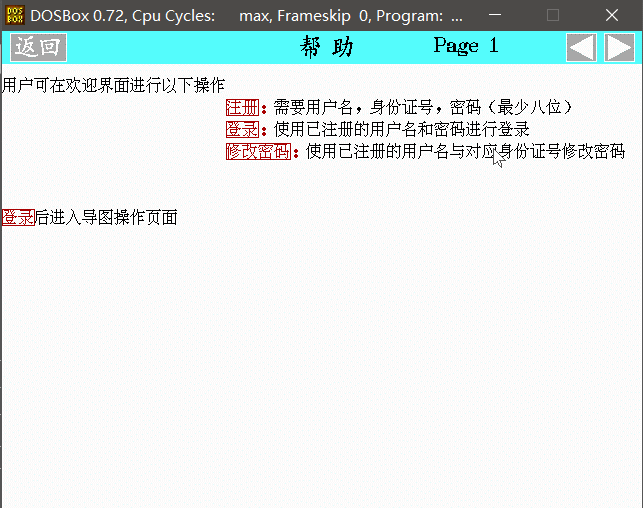
（1）进入打开页面，输入要打开的导图名称，点击【打开】可能会出现以下情况：

A.导图不存在，此时页面提示错误信息。

B.导图存在，此时跳回思维导图操作界面，并显示导图信息。

（2）点击【取消】，此时会跳回思维导图操作界面，并显示原来状态。

3.7 帮助界面



图表 10 帮助界面

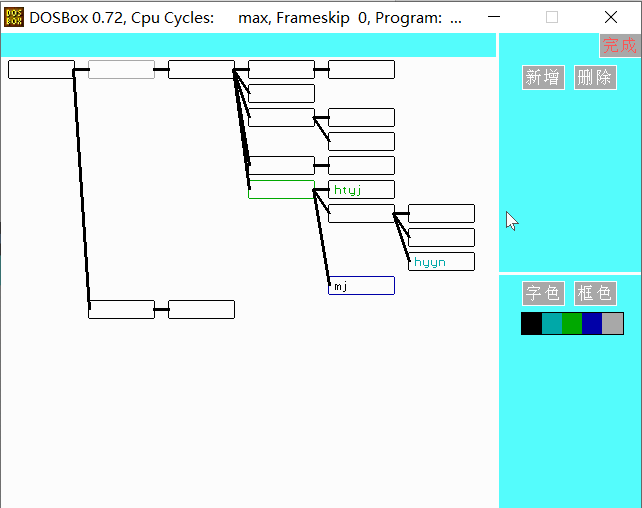
帮助界面由返回键，标题，页码，帮助文字信息，与翻页键组成，文字信息的按钮用红色标红。

帮助界面操作：

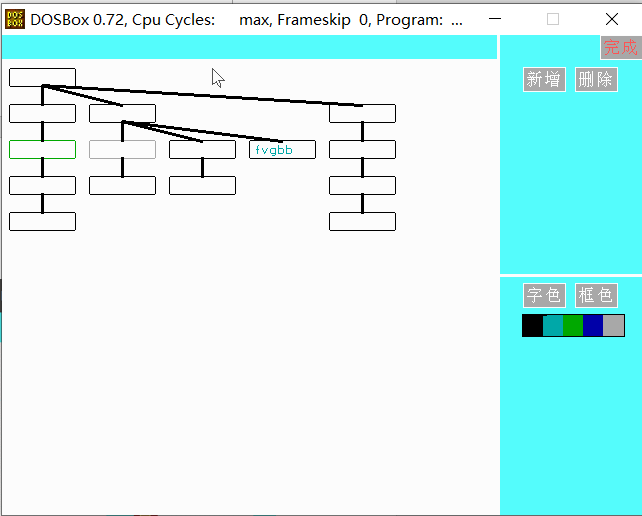
（1）点击翻页键可浏览帮助文字信息。

（2）点击【返回】可返回到欢迎界面。

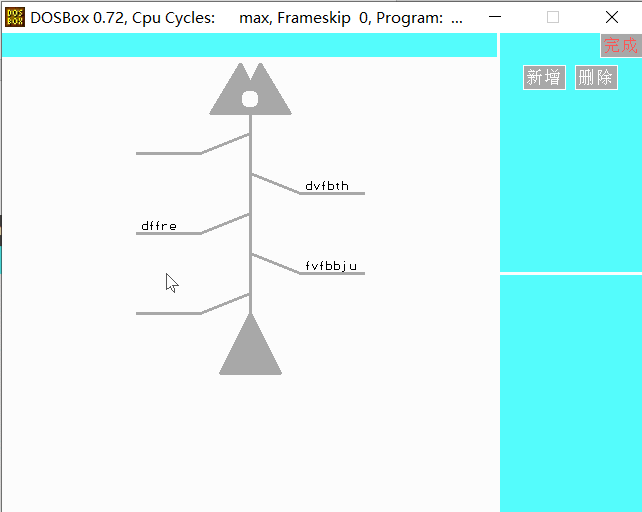
3.8 编辑界面



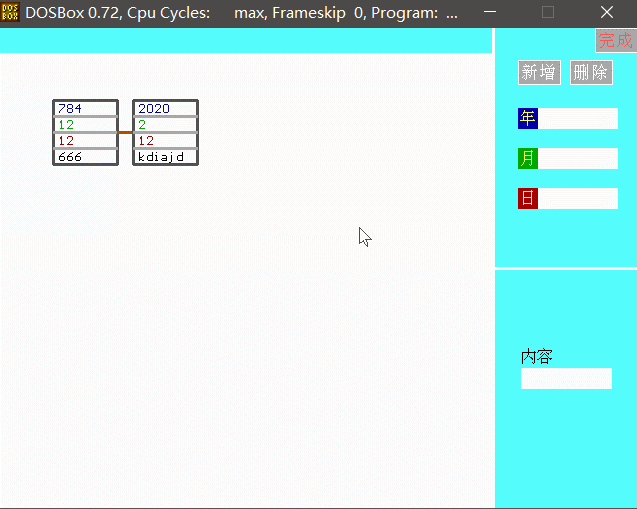
图表 11 单向思维导图编辑界面



图表 12 树状图编辑界面



图表 13 鱼骨图编辑界面



图表 14 时间轴编辑界面

编辑界面由主要导图显示信息，新增按钮，删除按钮与不同类型导图所用按钮组成。

编辑界面操作：

（1）单向思维导图：选中导图框后进入编辑文本状态，点击【回车】退出，之后可点击【新增】新增子代，【删除】删除框，【字色】根据当前选中的调色板中的颜色以及选中的框对框中字的颜色进行改变，【框色】根据当前选中的调色板中的颜色以及选中的框对框的颜色进行改变，【完成】退出编辑界面，回到操作界面。

（2）单向思维导图：选中导图框后进入编辑文本状态，点击【回车】退出，之后可点击【新增】新增子代，【删除】删除框，【字色】根据当前选中的调色板中的颜色以及选中的框对框中字的颜色进行改变，【框色】根据当前选中的调色板中的颜色以及选中的框对框的颜色进行改变，【完成】退出编辑界面，回到操作界面。

（3）鱼骨图：选中鱼骨后进入编辑文本状态，点击【回车】退出，之后可点击【新增】新增鱼骨，【删除】删除对应鱼骨，点击【完成】退出编辑界面，回到操作界面。

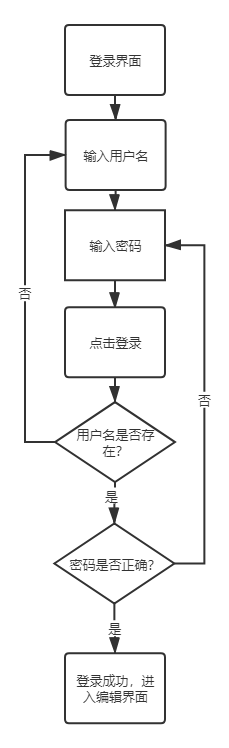
（4）时间轴：正确填写时间、内容信息后，点击【新增】新增一个时间框，会自动排序显示内容，选中时间框点击【删除】会将对应框删除，点击【完成】退出编辑界面，回到操作界面。

5．模块设计

5.1 登录模块

文件：hz.c EnterTs.c Log.c mouse.c

功能：输入用户名和密码，用户名必须存在，密码与用户名对应，点击登录进入编辑界面。

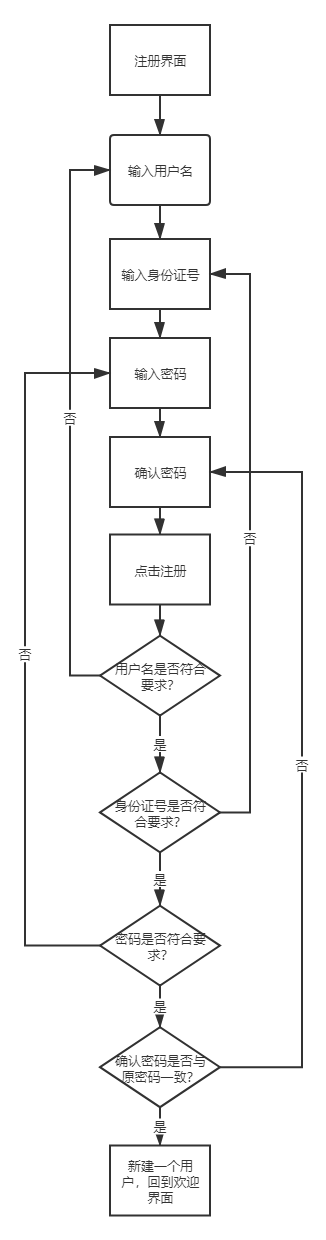


图表 15 登录流程

5.2 注册模块

文件：EnterT.c hz.c mouse.c signup.c

功能：输入用户名、身份证号、密码并确认密码，各部分必须符合要求，点击注册新建一个用户，并新建用户信息文件以及用户文件夹。

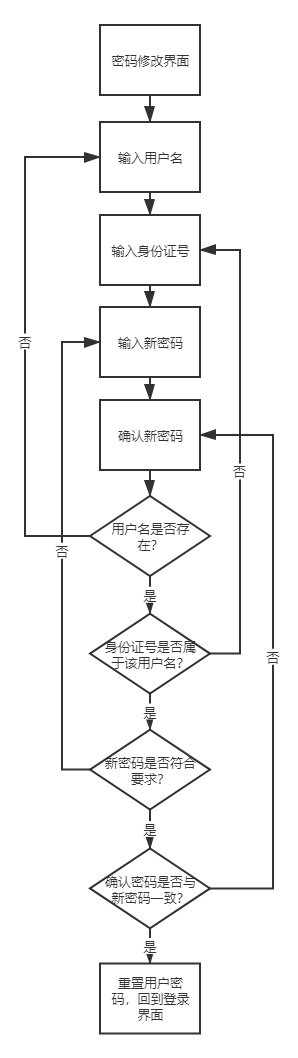


图表 16 注册流程

5.3 密码修改模块

文件：EnterT.c hz.c mouse.c Resetkey.c

功能：输入用户名、身份证号、密码并确认密码，各部分必须符合要求（身份证号与用户名相符已存在），点击确定重置密码。

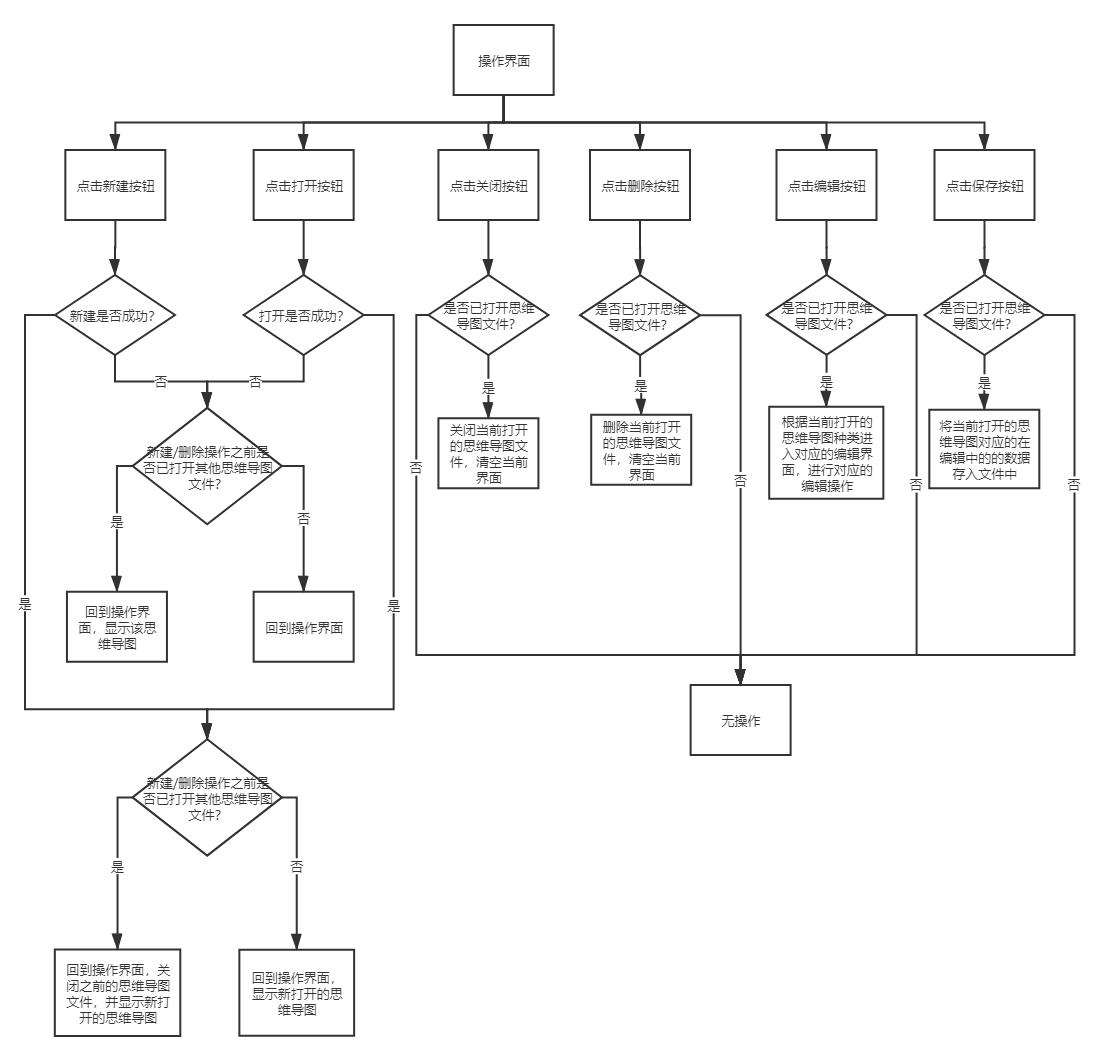


图表 17 重置密码流程

5.4 操作模块

文件：OperateT.c hz.c mouse.c Operate.c

功能：对思维导图进行新建，打开，关闭，删除，编辑，保存等操作。

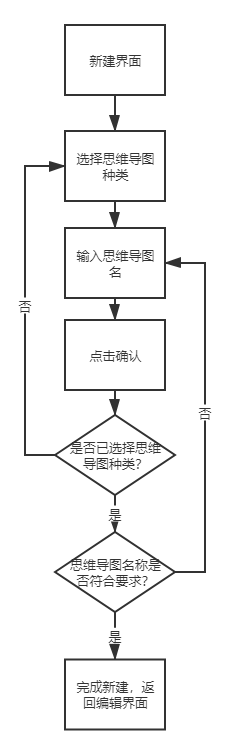


图表 18 操作思维导图界面流程

5.5 新建模块

文件：OperateT.c hz.c mouse.c newa.c

功能：点击选择思维导图的种类，并输入思维导图名称，完成新建。

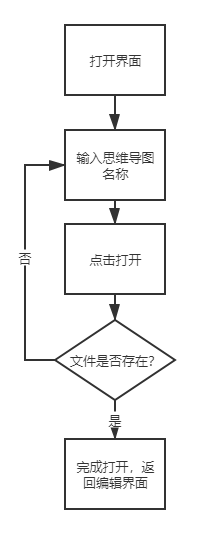


图表 19 新建流程

5.6 打开模块

文件：OperateT.c hz.c mouse.c Opena.c

功能：输入要打开的导图文件名，完成文件的打开。

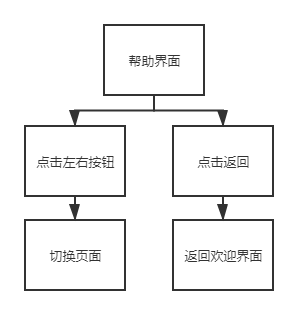


图表 20 打开思维导图流程

5.7 帮助模块

文件：Help.c hz.c mouse.c

功能：显示使用教程



图表 21 帮助界面流程

5.8 编辑模块

文件：Edit.c Edittr.c Editfb.c Edidate.c

Boxa.c Boxadl.c Boxtr.c Boxtrdl.c Boxfb.c Boxdate.c EditT.c

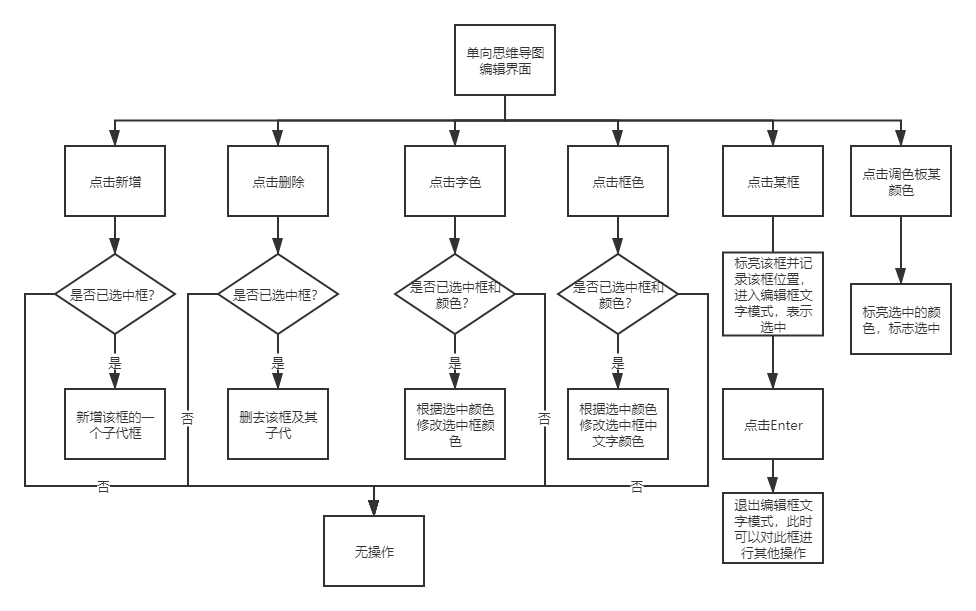
hz.c mouse.c

功能：编辑思维导图

5.8.1 编辑单向思维导图

文件：Edit.c Boxa.c Boxadl.c hz.c mouse.c

功能：单向思维导图的编辑

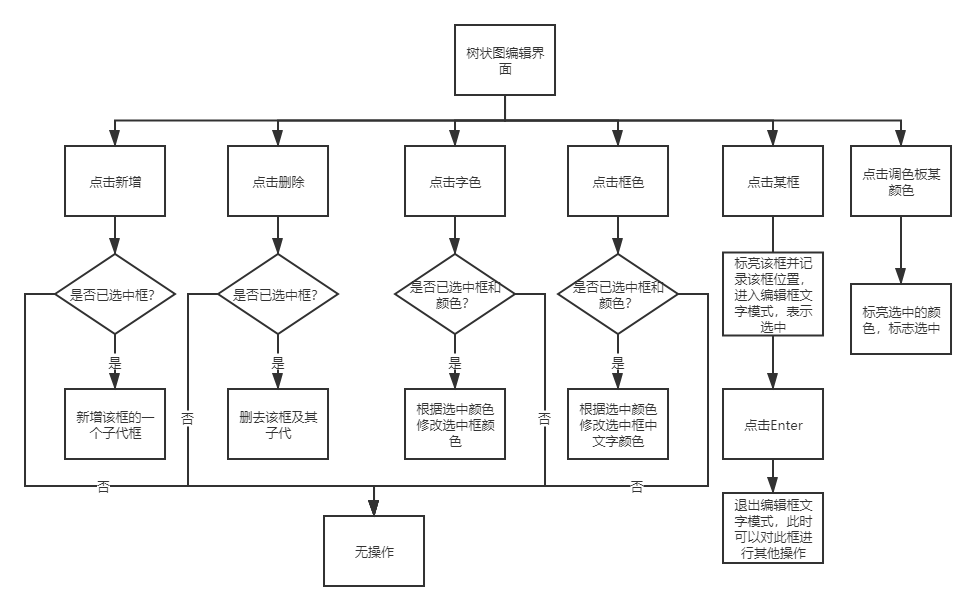


图表 22 编辑单向思维导图流程

5.8.2 编辑树状图

文件：Edittr.c Boxtr.c Boxtrdl.c hz.c mouse.c

功能：树状图的编辑

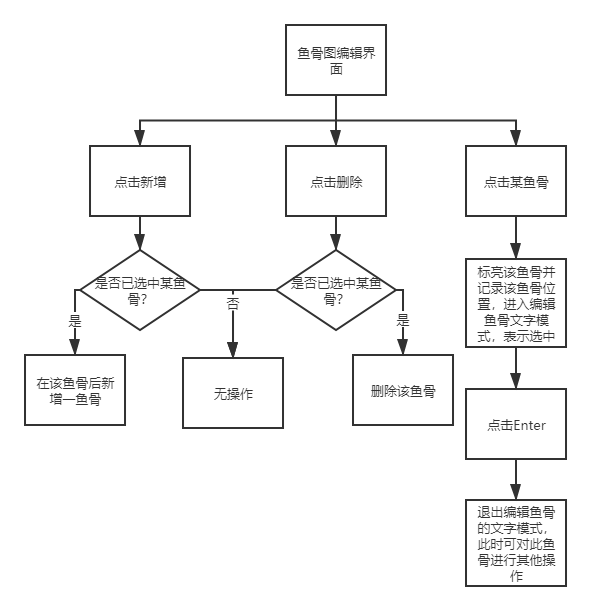


图表 23 编辑树状图流程

5.8.3 编辑鱼骨图

文件：Editfb.c Boxfb.c hz.c mouse.c

功能：鱼骨图的编辑

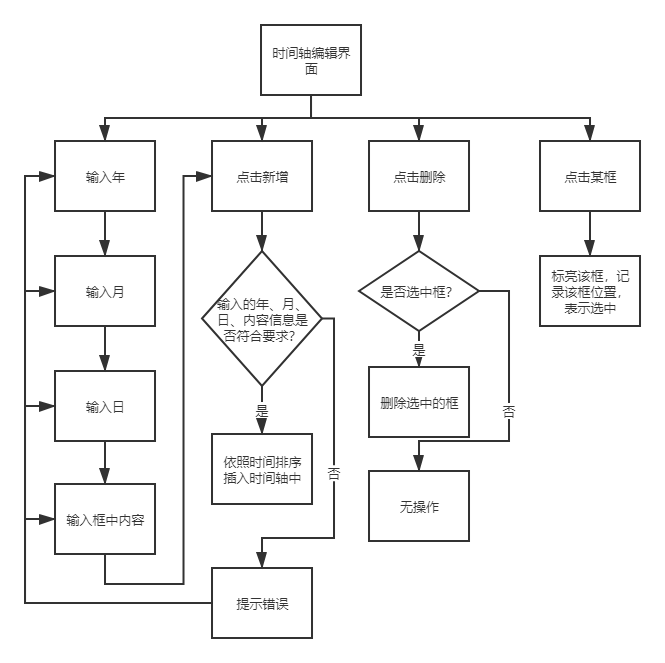


图表 24 编辑鱼骨图流程

5.8.4 编辑时间轴

文件：Edidate.c Boxdate.c hz.c mouse.c

功能：时间轴的编辑



图表 25 编辑时间轴流程

6．数据结构和数据流

#define **HORI** 6*//横向框上限数*

#define **VERT** 16*//纵向框上限数*

#define **VERTREE** 12*//树状图上限数*

#define **NON** 20*//未选中框的数*

#define **BONEMAX** 8*//鱼骨数*

#define **MAXDATE** 25*//时间轴最多数量*

typedef struct *//输入信息结构体*

{

    int x1;*//输入框坐标*

    int y1;

    int x2;

    int y2;

    char length;*//输入的最大长度*

    char cursor;*//光标位置*

    char flag;*//0代表不能输入，1代表可以输入*

    char str[19];*//输入的字符串内容*

} **ENTER**;

typedef struct *//用户相关信息*

{

    char username[12 + 1];*//用户名*

    char password[12 + 1];*//密码*

    char ID[18 + 1];*//身份证号*

} **USER**;

typedef struct *//思维导图文件信息*

{

    int type;*//思维导图的种类，1为单向思维导图，2为树状图*

    char mindname[8 + 1];*//思维导图名*

} **MIND**;

typedef struct

{

    int length;*//输入最大长度*

    int cursor;*//光标位置*

    char str[7 + 1];*//最长输入七个字符*

} **MINPUT**;

typedef struct *//思维导图框信息*

{

    int ste;*//框是否在使用中 0为不使用 1为使用*

    char content[8];*//框内文字*

*//届时思维导图框拟设置为6\*16的数组*

} **MINDBOX**;

typedef struct *//鱼骨图骨头信息*

{

    char content[8];*//骨头上的文字*

} **FISHBONE**;

typedef struct *//鱼骨图综合信息*

{

    FISHBONE bone[BONEMAX];*//骨头*

    int bonenum;*//骨头数*

} **FISHINFO**;

typedef struct *//时间轴单元信息*

{

    int year;*//年*

    int month;*//月*

    int day;*//日*

    char content[8];*//时间轴的文字*

} **DATEHAPPEN**;

typedef struct *//时间轴综合信息*

{

    DATEHAPPEN date[MAXDATE];*//某一天，天数暂定*

    int daynum;*//天数*

} **DATEINFO**;

ENTER：输入框信息，包含输入框左上角坐标x1,y1,右下角坐标x2,y2，输入长度上限length，光标位置cursor，是否能够输入标志flag及待输入文字内容str。其中光标在初始化时会置0，flag中0代表不能输入，1代表可以输入。

USER：用户信息结构体。其中包含用户名username，密码password，身份证号ID。其中用户名、身份证号、密码将在注册时被初始化，重置密码模块密码的信息可被修改，凭借每个人的身份证号。

MIND：思维导图信息结构体。包含思维导图类型type，思维导图名字mindname。其中type为1时是单向思维导图，为2时是树状图，为3时是鱼骨图，为4时是时间轴。

MINPUT：导图文本编辑信息结构体。包含输入最大长度length，输入光标位置cursor，内容str。在编辑思维导图页面传输编辑文本信息。

MINDBOX：单向思维导图及树状图框图结构体。包含框使用标志ste，框内文本content。其中ste为0意为框图未使用，为1时意为框图在使用。

FISHBONE：鱼骨文本信息结构体。包含文本内容content。

FISHINFO：鱼骨图综合信息结构体。包含鱼骨文本数组bone，骨头数bonenum，其中鱼骨文本数组有BONEMAX个。

DATEHAPPEN：时间轴单元信息结构体。包含年数year，月份month，天数day，时间轴文本内容content。其中年数合法范围为0－3000，月份合法范围为1－12，天数合法值要对应月及年。

DATEINFO：时间轴综合信息。包含时间轴单元信息数组date，时间轴单元数daynum。其中时间轴单元数组被声明为MAXDATE数量。

7．感想与体会

组员贾田旺的感想和反思：

C课设已经接近尾声，回望过去的一路，感叹自己的努力与成果的同时，我也有许多的话想说。

今年比较特殊，由于疫情的原因，许多课程的时间与考核方式都做了稍许改变。C课设也是如此，今年的C课设来的如此突然。在七月份末刚刚考完C语言的当天晚上，我们就进行了C课设选题宣讲，并进行了立刻选题。由于想挑战一下自我，并且一直对思维导图软件比较感兴趣，所以我选择了思维导图题目。

其实放假开始不久，我就进入了课设的准备阶段。首先我学习了课本后面上课时没有讲到的内容，如链表、文件操作、绘图的基础知识等，之后研读了C primer plus。慢慢的，我开始阅读老师们推荐的参考书，如C语言高级程序设计，王士元老师的参考书等，学习界面设计及编程的规范。虽然思维导图是一道新题，但是我也有浏览过学长以前的代码，汉字库与鼠标的相关知识我就是参考学长的代码与书本学习的。之后学习了一些数据结构的基础知识。

在基础知识准备完后，接下来就是完成思维导图软件的需求分析。我参考并使用了市面上许多主流的思维导图软件，如mindmaster、Xmind、mingmap等，再经过与队友的商量探讨，我们有了初步的想法。

课设正式开始是在七月中旬的时候，我至今依然记得我们画出欢迎界面的喜悦。欢迎、注册、登录、重置密码，慢慢的我们将课设的进度不断推前。但是不久，在中期验收前后，我们遇到了很多问题。编辑思维导图的核心算法由于bc的特性总是不能实现，我们也询问过许多学长与老师，但依旧无可奈何，我们的进度在九月中下旬一度停滞不前，我也一度懊恼过我为什么要选择一道新题来挑战自己。

难关的攻克是痛苦的，我和队友绞尽脑汁后终于想出来一种新的算法实现了导图的编辑。之后的路就走的顺利了一些，慢慢的我们的程序趋于完善。

最后说了这么多，我非常感谢我的队友。感谢他能够在这段课设的旅程中一直陪伴我。感谢学长及各位指导老师的答疑解惑。

组员倪启源的感想和反思：

回首这几个月的课设之旅，我的心中思绪万千：这一路，走的太不容易，但我们，也收获了很多……

记得当时考完C语言期末考试的当晚，我们就迎来了C语言的选题宣讲。20个选题，我看得眼花缭乱。而当晚又要立即给出选题的结果，最后经过一番略微讨论，我们选择了思维导图作为我们的选题。我们的C课设之旅就此开始了。

俗话说的好：万事开头难。一开始，我们意识到我们知识储备上的不足，开始大量地进行学习。图形，文件，数据结构……我们浏览了大量的资料。但是话说回来，纸上得来终觉浅。当我们真正计划开始进行代码的编写时，才发现并没有那么简单，虽有学长老师的耐心指导，但很多操作还是得一点一点摸索。在这里，我衷心地感谢我的队友。不得不承认的是，在初期的知识准备以及一些代码编写上，我的进度非常的缓慢，甚至一度出现了停滞不前的现象。是我的队友，给予了我很多的帮助，一步一步的通过网络聊天等方式对我进行指导，带我渡过了开头最艰难的时期……

后来登录、注册、修改密码等功能都实现了，我们进入了核心功能实现的初级阶段。新建思维导图，打开思维导图……看似简单的功能，在实现的时候，也着实是费脑筋。由于我们当时对于文件的一系列操作还不太熟练，因此也经历了一段相对比较艰难的时光……

此后，我们开始了核心代码的实现阶段。也就是思维导图的新增、删除、文字编辑等功能的实现。当时的我们，还没有意识到，我们将迎来一段代码编写的瓶颈期。我们尝试了一套又一套的方案，却总是由于各种原因而不得不舍弃。那一段时间，我们都睡不好。每天就想着如何进行核心功能的实现，最终验收的日子一天一天走近，我们也一天一天变得着急……

几次方案的失败，让我们不得不想到：我们，是否囿于一种思维定势之中了？回想之前两次失败的方案，大量的递归嵌套……是否存在一种方式，可以不需要大量的递归嵌套？

推翻旧有的思维定势，我们开始转换看待问题的角度。我们把思维导图单纯地看作是一张由许多框和线组成的一个图形。我们要做的新增删除，其实也就是将对应的框进行位置的变换，或者让对应的框消失。循着这个思路，我们开始了代码的编写……由于这种方案并不涉及递归操作，因此对运行内存的需求也就较小……终于，在一个夜晚，我们成功实现了新增操作的函数编写，我至今都还记得我们当时的激动之情。虽然有些小bug，但是我们已经看到了曙光……

此后我们又进行了其他函数的编写并进行了一定的调试，又新创了几种思维导图的种类……我们的程序，日趋完善。

勇于推翻当前固有的思维定势，是需要勇气的。我很庆幸我们做到了。

最后，我深深地感谢我的队友。在这几个月的时间里，他给予了我充分的支持和帮助。同时，我也要感谢耐心指导帮助我们的老师和同学们。

8．出现的问题及解决办法

8.1 用户的文件管理问题

在进行用户对应的思维导图文件的管理时，我们本想将所有思维导图文件都存放在一个文件夹内，这样，在打开时，只需找到对应的文件名的文件即可打开对应的思维导图文件。但是，我们发现，这样做有一个弊端：不同的用户不可能拥有相同名字的文件。因为两个文件在同一个文件夹下时，这两个文件的文件名必定不相同。

为解决这一问题，我们一开始考虑将文件名设置为“用户名+导图名”的形式。然而，这种方法事实上是不太适用的。因为在Borlandc中，文件名最长只能有8个字符，这意味着上述形式会严重影响用户名和导图名的长度，进而影响用户的使用体验。事实上，我们想到了更好的方法：每注册一个新用户后，就为这个用户设置一个文件夹，用于存放该用户的思维导图文件。经过查阅资料，我们找到了设置文件夹对应的函数，即mkdir函数。通过这个函数，我们实现了将思维导图根据用户的不同分类存储，使得文件管理的条理更加清晰。

8.2 思维导图核心功能的实现问题

在进行单向思维导图以及树状图核心功能实现代码的书写时，我们尝试了至少种方法。最终采用了目前的这种方法，现就尝试的方法中比较有代表性的几种进行叙述。

8.2.1 通过数据结构实现（利用一维数组的下标实现父框与子框之间逻辑关系）

一开始，我们参考了数据结构的书籍，考虑用类似于树的数据结构对思维导图的信息进行存储。即将每一个思维导图框节点设置为一个结构体，里面用指针存储其父节点以及子代节点的位置，并存储节点本身的一些信息，譬如框内的文字，框的颜色，文字的颜色，框的位置等。待到有一定思路时，我们开始着手进行代码的实现。但不久我们就发现一个棘手的问题：如果我们用每一个结构体存储其父框以及字框的位置指针，那么在程序关闭时，这些地址指针对应的内存空间的信息就会从内存中被清除。这意味着指针所存储的信息失去了其意义。所以，在对思维导图信息进行存储时，用指针来存储框与框之间的逻辑关系的做法显然是不太合适的。

因此，我们考虑设置一个一维的结构数组，这个结构数组存储着当前思维导图已新增的所有框的信息。而之前所考虑的地址，则改作用一维数组中对应结构元素的下标进行存储。这样，要想找到某个框对应的父框和子框，只需要通过框元素对应的下标即可找到。同时，在进行思维导图的存储时，只需要将此数组存入文件即可，将数组读入和读出时对应的下标处所存储的元素信息不会受到破坏。

我们先尝试做了删除代码，进行了大量的debug工作。当时为实现一个删除框操作我们写了将近10个工具函数。由于删除需要遍历对应被删除框的所有子框，同时在删除后这些子框的被删除意味着在一维数组一些未被删除的框元素的位置需要进行移动，对删除后留下的空档进行填补。这意味着这些框元素的下标也要随之而发生改变。与此同时，这些框所存储的父子框元素的下标也要进行相应的改变。除此之外，还有一些框的坐标需要进行改变……这些改变意味着大量的递归调用和嵌套。最终，deletebox函数在debug情况下正常运行，但是当我们对进行run操作时，发现程序运行出现了崩溃。在询问了老师和同学后，我们得出结论：大量的嵌套和递归导致了这次的崩溃。一开始在询问了老师之后，我们尝试将BC中Code Generation的Model设置改为Huge模式。但是函数运行依然出现了崩溃，我们又根据老师的建议试着将一些指针改为far指针。但后来我们突然想到：如此庞大而复杂的功能代码的实现，涉及如此之多的嵌套递归，恐怕huge也不足以支持，更何况这只是单纯的一步删除某一框的操作。于是，我们萌生了重新设计思路的想法。

8.2.2 通过数据结构实现（利用指针实现父框与子框之间逻辑关系）

我们后来也考虑过就单纯地使用地址索引来表示框与框的逻辑关系这一做法，只是在文件保存的时候以第一种思路中的一维数组的方式进行保存，只需在读入和读出文件的时候进行两种存储方式之间的相互转换即可。但是，这种方法事实上依然包含着大量的递归与嵌套，因此在尝试了一两天之后我们放弃了这种做法。

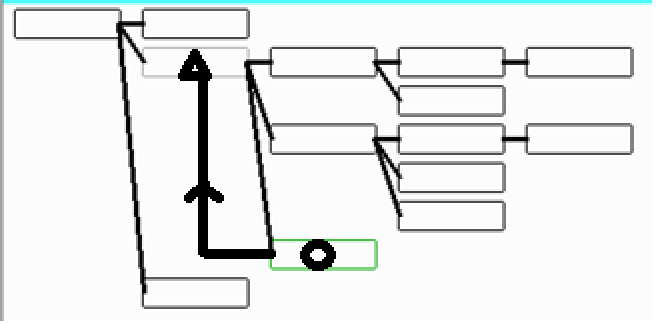
8.2.3 通过框的位置关系实现（利用框的位置关系反推父框与子框逻辑关系）

那段时间我们一直没有进展，我们一直找不出合适的实现方式。后来，我们突然发现，有时候一个问题太复杂，可能是由于我们把问题想得太复杂了。为什么我们一定要让框与框之间存在逻辑关系？我们对框的新增删除，实质上也就意味着框的出现与消失，以及框的位置的改变罢了。

这是否意味着，我们只需要摸清新增删除框后框的位置变化规律以及框的出现消失规律，并将对应的框进行连线，就可以实现新增和删除的算法了？其实，这也就意味着我们无需将思维导图看作一个有着逻辑关系的“树”状结构，而只需要将其看作是一张由许多框和线组成的图形就可以了。

我们顺着这个思路，将编辑区域分成了6×16个部分，这可以用一个二维数组来实现，二维数组的一个元素对应区域中的一个部分。每一个部分都只能存储一个框的信息。我们用了一个ste元素来表示某一个部分是否存储着一个框（ste==1表示该部分存在框，ste==0表示该部分无框）。如果进行了删除操作，则将对应被删除的框的ste置为0，同时将一部分框的位置进行移动，也就是将一部分框的信息赋值给另一部分框。只需通过一定的循环判断操作判断当前框关联的框的位置摆放情况，再根据不同的摆放情况来进行对应的移动框操作即可实现新增和删除操作。

值得一提的是，我们发现在我们设置的这种编辑模式中，要想找到父框，只需要从对应的框左侧一列与本框纵坐标相同的位置开始找起，依次向上遍历，直至找到ste==1的部分即可，如图表26中的⚪框，要想找到其父框，只需依照图中的路径即可找到。即图中的Δ框。



图表 26 寻找某框的父框

从图表26中也可以看出，我们的框的位置具有一定的规律性，事实上，我们将框的位置的移动转化成为二维数组中元素的信息的相互转移，将清除框信息的操作转化成了二维数组中某一元素的信息的清空同时将ste置为0。而画出思维导图，事实上也就是遍历二维数组，将ste==1的元素对应的信息画出来即可。至于连线的操作，也就是从子框寻觅到父框，然后将二者相连即可。

8.3 “最后一个框”问题

在进行单向思维导图、树状图、鱼骨图的编辑时，如果当前页面仅剩一个框（鱼骨），删去后我们将无法继续执行其他操作，因为所有操作（包括新增操作）都是基于已有的框（鱼骨）而实现的。因此我们增设了一个判断机制，当判定当前页面仅剩一个框（鱼骨）时，将无法执行删除操作。

9．源代码

9.1 welcome.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  完成欢迎界面及其功能*

*Function list :  welcome    欢迎界面功能*

*welcome\_screen    绘制欢迎界面*

*Attention:*

*Author: 贾田旺*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "welcome.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: welcome*

*Function：欢迎界面相关信息与跳转功能*

*Calls: 无*

*Called By: main*

*Parameter： 无*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **welcome**(void)

{

    int page = 1;

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值*

*//0 无操作状态 1 退出键 2 注册键 3 登陆键*

*//4 帮助键*

**clrmous**(MouseX, MouseY);

**delay**(100);

*//mouseinit();*

**welcome\_screen**();

**mouseinit**();

    while (page == 1)

    {

**newmouse**(&MouseX, &MouseY, &press);

*//退出框*

        if (**mouse\_press**(590, 0, 640, 35) == 2) *//在退出框且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

**setcolor**(RED);

**setlinestyle**(0, 0, 3);

**rectangle**(590, 0, 640, 35);

            }

        }

        else if (**mouse\_press**(590, 0, 640, 35) == 1) *//在退出框且点击*

        {

            page = 0;

        }

*//注册框*

        else if (**mouse\_press**(240, 300, 400, 340) == 2) *//在注册框未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

**setcolor**(LIGHTMAGENTA);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTMAGENTA);

**rectangle**(240, 300, 400, 340);

**fillellipse**(252, 320, 9, 9);

**puthz**(275, 310, "用户注册", 24, 32, LIGHTMAGENTA);

**setlinestyle**(0, 0, 3);

**line**(195, 320, 240, 320);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillellipse**(120, 320, 75, 26);

**puthz**(53, 312, "点击即可加入我们", 16, 17, LIGHTMAGENTA);

            }

        }

        else if (**mouse\_press**(240, 300, 400, 340) == 1) *//在注册框且点击*

        {

            page = 2;

        }

*//登录框*

        else if (**mouse\_press**(240, 350, 400, 390) == 2) *//在登录框未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

**setcolor**(LIGHTMAGENTA);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTMAGENTA);

**rectangle**(240, 350, 400, 390);

**fillellipse**(252, 370, 9, 9);

**puthz**(275, 360, "用户登录", 24, 32, LIGHTMAGENTA);

**setlinestyle**(0, 0, 3);

**line**(445, 370, 400, 370);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillellipse**(520, 370, 75, 26);

**puthz**(453, 362, "登录即可纵情表达", 16, 17, LIGHTMAGENTA);

            }

        }

        else if (**mouse\_press**(240, 350, 400, 390) == 1)

        {

            page = 3;

        }

        else if (**mouse\_press**(0, 0, 50, 35) == 2) *//帮助未点击*

        {

            pre\_state = state;

            state = 4;

            if (pre\_state != 4) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 3);

**rectangle**(0, 0, 50, 35);

            }

        }

        else if (**mouse\_press**(0, 0, 50, 35) == 1) *//帮助且点击*

        {

            page = 6;

        }

*//无操作状态*

        else

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**rectangle**(590, 0, 640, 35);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, WHITE);

**rectangle**(240, 300, 400, 340);

**fillellipse**(252, 320, 9, 9);

**puthz**(275, 310, "用户注册", 24, 32, WHITE);

**setlinestyle**(0, 0, 3);

**line**(195, 320, 240, 320);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillellipse**(120, 320, 75, 26);

**puthz**(53, 312, "快来写下你的想法", 16, 17, LIGHTBLUE);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, WHITE);

**rectangle**(240, 350, 400, 390);

**fillellipse**(252, 370, 9, 9);

**puthz**(275, 360, "用户登录", 24, 32, WHITE);

**setlinestyle**(0, 0, 3);

**line**(445, 370, 400, 370);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillellipse**(520, 370, 75, 26);

**puthz**(453, 362, "快来架构你的思维", 16, 17, LIGHTBLUE);

                break;

            case 4:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**rectangle**(0, 0, 50, 35);

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: welcome\_screen*

*Function：画欢迎界面*

*Calls: 无*

*Called By: welcome*

*Parameter：无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **welcome\_screen**(void)

{

*//用于画m字母的数组*

    int a[32] = {250, 170, 270, 60, 300, 60, 320, 140, 340, 60, 370, 60, 390, 170, 380, 170, 362, 80, 348, 80, 325, 170, 315, 170, 292, 80,

                 278, 80, 260, 170, 250, 170};

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//退出按钮*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(590, 0, 640, 35); *//退出按钮*

**rectangle**(590, 0, 640, 35);

**puthz**(600, 10, "退出", 12, 18, WHITE);

*//帮助按钮*

**bar**(0, 0, 50, 35); *//退出按钮*

**rectangle**(0, 0, 50, 35);

**puthz**(10, 10, "帮助", 12, 18, WHITE);

*//画M logo*

**setfillstyle**(SOLID\_FILL, YELLOW);

**setlinestyle**(0, 0, 1);

**fillellipse**(320, 120, 100, 100);

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**fillellipse**(320, 120, 90, 90);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillpoly**(16, a);

*//思维导图汉字*

**puthz**(225, 235, "思维导图", 48, 50, LIGHTBLUE);

*//注册按钮*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(240, 300, 400, 340);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(240, 300, 400, 340);

**puthz**(275, 310, "用户注册", 24, 32, WHITE);

*//登录按钮*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(240, 350, 400, 390);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(240, 350, 400, 390);

**puthz**(275, 360, "用户登录", 24, 32, WHITE);

*//两个按钮的修饰*

**setfillstyle**(SOLID\_FILL, WHITE);

**fillellipse**(252, 370, 9, 9);

**fillellipse**(252, 320, 9, 9);

**setfillstyle**(SOLID\_FILL, YELLOW);

**fillellipse**(520, 370, 75, 26);

**fillellipse**(120, 320, 75, 26);

**puthz**(453, 362, "快来架构你的思维", 16, 17, LIGHTBLUE);

**puthz**(53, 312, "快来写下你的想法", 16, 17, LIGHTBLUE);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**line**(445, 370, 400, 370);

**line**(195, 320, 240, 320);

*//版权信息*

**setcolor**(WHITE);

**settextstyle**(0, 0, 1);

**outtextxy**(260, 470, "Copyright C 2020 HUST-Jiatianwang, Niqiyuan.");

}

9.2 Signup.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  注册页面及函数*

*Function list : sign\_up 注册函数*

*sign\_up\_screen  绘制注册页面*

*Attention:*

*Author: 贾田旺*

*History: 1 完成页绘制及跳转 2 完成注册功能*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Signup.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: sign\_up*

*Function：注册函数*

*Calls: sign\_up\_screen*

*input\_info*

*judge\_samename*

*judge\_ID*

*judge\_password*

*write\_info*

*new\_file*

*Called By: main*

*Parameter: 无*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **sign\_up**(void)

{

    int page = 2;

    int state = 0, pre\_state = 0;                                     *//表示鼠标状态的相关值*

*// 0 无操作状态 1 退出按钮 2 返回按钮 3 四个输入白框*

*// 4 注册按钮*

    ENTER username = {96 + 128, 96, 96 + 448, 96 + 32, 8, 0, 0, ""}; *//ENTER为输入信息结构体，详见data.h*

    ENTER ID = {96 + 128, 160, 96 + 448, 160 + 32, 18, 0, 0, ""};

    ENTER password1 = {96 + 128, 224, 96 + 448, 224 + 32, 12, 0, 0, ""};

    ENTER password2 = {96 + 128, 288, 96 + 448, 288 + 32, 12, 0, 0, ""};

**clrmous**(MouseX, MouseY); *//清除鼠标，画原背景，及释放原背景内存，避免留痕*

**delay**(100);

**sign\_up\_screen**();

**mouseinit**(); *//初始化*

    while (page == 2)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (username.flag != 1 && ID.flag != 1 && password1.flag != 1 && password2.flag != 1)

        {

            while (**bioskey**(1))

            {

**getch**();

            }

        }

**input\_info**(96 + 128 + 4, 96 + 4, 7, 1, &username);

**input\_info**(96 + 128 + 4, 160 + 4, 7, 1, &ID);

**input\_info**(96 + 128 + 4, 224 + 4, 7, 0, &password1);

**input\_info**(96 + 128 + 4, 288 + 4, 7, 0, &password2);

        if (**mouse\_press**(608, 0, 640, 32) == 2) *//退出框且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

            }

        }

        else if (**mouse\_press**(608, 0, 640, 32) == 1) *//退出框且点击*

        {

            page = 0;

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 2) *//返回框且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**rectangle**(0, 0, 32, 32);

            }

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 1) *//返回框且点击*

        {

            page = 1;

        }

        else if (**mouse\_press**(96 + 128, 96, 96 + 448, 96 + 32) == 2 || **mouse\_press**(96 + 128, 160, 96 + 448, 160 + 32) == 2 ||

**mouse\_press**(96 + 128, 224, 96 + 448, 224 + 32) == 2 || **mouse\_press**(96 + 128, 288, 96 + 448, 288 + 32) == 2)

*//四个输入白框 且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 2; *//光标*

            }

        }

        else if (**mouse\_press**(256, 352, 256 + 128, 352 + 48) == 2) *//注册按钮且未点击*

        {

            pre\_state = state;

            state = 4;

            if (pre\_state != 4) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(268, 356, 268 + 112, 356 + 32);

**puthz**(264 + 20, 352 + 8, "注册", 24, 56, WHITE);

            }

        }

        else if (**mouse\_press**(256, 352, 256 + 128, 352 + 48) == 1) *//注册按钮且点击*

        {

*//清除提示信息*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(96 + 448 - 16 \* 6, 96 + 32 + 4, 96 + 448, 96 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 8, 160 + 32 + 4, 96 + 448, 160 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 9, 224 + 32 + 4, 96 + 448, 224 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 9, 288 + 32 + 4, 96 + 448, 288 + 32 + 4 + 16);

*//确认是否输入完全*

            if (username.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 6, 96 + 32 + 4, "未输入用户名", 16, 16, RED);

            }

            else if (ID.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 7, 160 + 32 + 4, "未输入身份证号", 16, 16, RED);

            }

            else if (password1.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 5, 224 + 32 + 4, "未输入密码", 16, 16, RED);

            }

            else if (password2.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 6, 288 + 32 + 4, "未确认密码", 16, 16, RED);

            }

*//判断用户名是否重复*

            else if (**judge\_samename**(username.str))

            {

**puthz**(96 + 448 - 16 \* 6, 96 + 32 + 4, "用户名已存在", 16, 16, RED);

            }

*//判断身份证号是否符合要求*

            else if (!**judge\_ID**(ID.str))

            {

**puthz**(96 + 448 - 16 \* 8, 160 + 32 + 4, "身份证号格式错误", 16, 16, RED);

            }

*//判断密码是否符合要求*

            else if (!**judge\_password**(password1.str))

            {

**puthz**(96 + 448 - 16 \* 9, 224 + 32 + 4, "密码长度至少为八位", 16, 16, RED);

            }

*//判断两次输入密码是否相同*

            else if (**strcmp**(password1.str, password2.str))

            {

**puthz**(96 + 448 - 16 \* 9, 288 + 32 + 4, "两次输入密码不一致", 16, 16, RED);

            }

*//写入注册信息,新建账户文件*

            else

            {

**write\_info**(username.str, ID.str, password1.str);

**new\_file**(username.str);

**newdrt**(username.str);

                page = 1; *//返回主界面*

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(0, 0, 32, 32);

                break;

            case 3:

                MouseS = 0;

                break;

            case 4:

                MouseS = 0;

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**puthz**(264 + 20, 352 + 8, "注册", 24, 56, WHITE);

                break;

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: sign\_up\_screen*

*Function：绘制注册界面*

*Calls: 无*

*Called By: sign\_up*

*Parameter：无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **sign\_up\_screen**(void)

{

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//退出程序的按钮*

**setcolor**(LIGHTGRAY);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(608, 0, 640, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

*//返回按钮*

**setcolor**(WHITE);

**setcolor**(LIGHTGRAY);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(0, 0, 32, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(0, 0, 32, 32);

**setlinestyle**(0, 0, 3);

**line**(24, 4, 8, 16);

**line**(8, 16, 24, 28);

*//背景*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**rectangle**(56, 64, 582, 416);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(56, 64, 582, 416);

*//用户名框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 96, 96 + 128, 96 + 32);

**puthz**(96 + 4, 96 + 4, "用户名", 24, 48, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 96, 96 + 448, 96 + 32);

*//身份证号框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 160, 96 + 128, 160 + 32);

**puthz**(96 + 4, 160 + 4, "身份证号", 24, 32, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 160, 96 + 448, 160 + 32);

*//密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 224, 96 + 128, 224 + 32);

**puthz**(96 + 20, 224 + 4, "密码", 24, 64, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 224, 96 + 448, 224 + 32);

*//确认密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 288, 96 + 128, 288 + 32);

**puthz**(96 + 4, 288 + 4, "确认密码", 24, 32, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 288, 96 + 448, 288 + 32);

*//注册按钮*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**puthz**(264 + 20, 352 + 8, "注册", 24, 56, WHITE);

}

9.3 ResetKey.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  完成重置密码页面及相关功能*

*Function list :  reset\_key    重置密码函数*

*reset\_key\_screen    绘制重置密码页面*

*Attention:*

*Author: 贾田旺*

*History: 1 完成页面绘制及跳转 2 完成具体重置密码功能*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "ResetKey.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: reset\_key*

*Function：重置密码函数*

*Calls:  reset\_key\_screen*

*input\_info*

*judge\_samename*

*right\_ID*

*judge\_password*

*rewrite\_key*

*Called By: main*

*Parameter: 无*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **reset\_key**(void)

{

    int page = 4;

    int state = 0, pre\_state = 0;                                    *//表示鼠标状态的相关值*

*// 0 无操作状态 1 退出按钮 2 返回按钮 3 四个输入白框*

*// 4 确定按钮*

    ENTER username = {96 + 128, 96, 96 + 448, 96 + 32, 8, 0, 0, ""}; *//ENTER为输入信息结构体，详见data.h*

    ENTER ID = {96 + 128, 160, 96 + 448, 160 + 32, 18, 0, 0, ""};

    ENTER password1 = {96 + 128, 224, 96 + 448, 224 + 32, 12, 0, 0, ""};

    ENTER password2 = {96 + 128, 288, 96 + 448, 288 + 32, 12, 0, 0, ""};

**clrmous**(MouseX, MouseY);

**delay**(100);

**reset\_key\_screen**();

**mouseinit**();

    while (page == 4)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (username.flag != 1 && ID.flag != 1 && password1.flag != 1 && password2.flag != 1)

        {

            while (**bioskey**(1))

            {

**getch**();

            }

        }

**input\_info**(96 + 128 + 4, 96 + 4, 7, 1, &username);

**input\_info**(96 + 128 + 4, 160 + 4, 7, 1, &ID);

**input\_info**(96 + 128 + 4, 224 + 4, 7, 0, &password1);

**input\_info**(96 + 128 + 4, 288 + 4, 7, 0, &password2);

        if (**mouse\_press**(608, 0, 640, 32) == 2) *//退出框且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

            }

        }

        else if (**mouse\_press**(608, 0, 640, 32) == 1) *//退出框且点击*

        {

            page = 0;

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 2) *//返回框且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**rectangle**(0, 0, 32, 32);

            }

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 1) *//返回框且点击*

        {

            page = 3;

        }

        else if (**mouse\_press**(96 + 128, 96, 96 + 448, 96 + 32) == 2 || **mouse\_press**(96 + 128, 160, 96 + 448, 160 + 32) == 2 ||

**mouse\_press**(96 + 128, 224, 96 + 448, 224 + 32) == 2 || **mouse\_press**(96 + 128, 288, 96 + 448, 288 + 32) == 2)

*//四个输入白框 且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 2; *//光标*

            }

        }

        else if (**mouse\_press**(256, 352, 256 + 128, 352 + 48) == 2) *//确定按钮且未点击*

        {

            pre\_state = state;

            state = 4;

            if (pre\_state != 4) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(268, 356, 268 + 112, 356 + 32);

**puthz**(264 + 20, 352 + 8, "确定", 24, 56, WHITE);

            }

        }

        else if (**mouse\_press**(256, 352, 256 + 128, 352 + 48) == 1) *//确定按钮且点击*

        {

*//清除提示信息*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(96 + 448 - 16 \* 6, 96 + 32 + 4, 96 + 448, 96 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 7, 160 + 32 + 4, 96 + 448, 160 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 9, 224 + 32 + 4, 96 + 448, 224 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 9, 288 + 32 + 4, 96 + 448, 288 + 32 + 4 + 16);

*//确认是否输入完全*

            if (username.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 6, 96 + 32 + 4, "未输入用户名", 16, 16, RED);

            }

            else if (ID.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 7, 160 + 32 + 4, "未输入身份证号", 16, 16, RED);

            }

            else if (password1.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 5, 224 + 32 + 4, "未输入密码", 16, 16, RED);

            }

            else if (password2.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 6, 288 + 32 + 4, "未确认密码", 16, 16, RED);

            }

*//判断是否存在用户名*

            else if (!**judge\_samename**(username.str))

            {

**puthz**(96 + 448 - 16 \* 6, 96 + 32 + 4, "用户名不存在", 16, 16, RED);

            }

*//判断身份证号是否正确*

            else if (!**right\_ID**(username.str, ID.str))

            {

**puthz**(96 + 448 - 16 \* 6, 160 + 32 + 4, "身份证号错误", 16, 16, RED);

            }

*//判断密码是否符合要求*

            else if (!**judge\_password**(password1.str))

            {

**puthz**(96 + 448 - 16 \* 9, 224 + 32 + 4, "密码长度至少为八位", 16, 16, RED);

            }

*//判断两次输入密码是否相同*

            else if (**strcmp**(password1.str, password2.str))

            {

**puthz**(96 + 448 - 16 \* 9, 288 + 32 + 4, "两次输入密码不一致", 16, 16, RED);

            }

            else *//操作成功写入新密码，返回登录页面*

            {

**rewrite\_key**(username.str, password1.str);

                page = 3;

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**rectangle**(0, 0, 32, 32);

                break;

            case 3:

                MouseS = 0;

                break;

            case 4:

                MouseS = 0;

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**puthz**(264 + 20, 352 + 8, "确定", 24, 56, WHITE);

                break;

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: reset\_key\_screen*

*Function：绘制重置页面*

*Calls: 无*

*Called By: reset\_key*

*Parameter: 无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **reset\_key\_screen**(void)

{

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//退出程序的按钮*

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(608, 0, 640, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

*//返回按钮*

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(0, 0, 32, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(0, 0, 32, 32);

**setlinestyle**(0, 0, 3);

**line**(24, 4, 8, 16);

**line**(8, 16, 24, 28);

*//背景*

**setlinestyle**(0, 0, 3);

**setcolor**(WHITE);

**rectangle**(56, 64, 582, 416);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(56, 64, 582, 416);

*//用户名框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 96, 96 + 128, 96 + 32);

**puthz**(96 + 4, 96 + 4, "用户名", 24, 48, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 96, 96 + 448, 96 + 32);

*//身份证号框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 160, 96 + 128, 160 + 32);

**puthz**(96 + 4, 160 + 4, "身份证号", 24, 32, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 160, 96 + 448, 160 + 32);

*//新密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 224, 96 + 128, 224 + 32);

**puthz**(96 + 4, 224 + 4, "新密码", 24, 48, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 224, 96 + 448, 224 + 32);

*//确认密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 288, 96 + 128, 288 + 32);

**puthz**(96 + 4, 288 + 4, "确认密码", 24, 32, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 288, 96 + 448, 288 + 32);

*//确定按钮*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(264, 352, 264 + 120, 352 + 40);

**puthz**(264 + 20, 352 + 8, "确定", 24, 56, WHITE);

}

9.4 Log.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  登录页面及相关功能*

*Function list :  log    登录函数*

*log\_screen  绘制登录页面*

*Attention:*

*Author: 贾田旺*

*History: 1 完成绘制及跳转  2  完成登录功能*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Log.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: log*

*Function：登录函数*

*Calls: input\_info*

*judge\_samename*

*right\_key*

*Called By: main*

*Parameter: account\_name  用户名 登录后用来保存与用户相关的文件*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **log**(char \*account\_name)

{

    int page = 3;

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态 1 退出按钮 2 返回按钮 3 登录按钮*

*// 4 忘记密码按钮 5 两个输入白框*

    ENTER username = {96 + 128, 128, 96 + 448, 128 + 32, 8, 0, 0, ""};

    ENTER password = {96 + 128, 188, 96 + 448, 188 + 32, 12, 0, 0, ""};

**clrmous**(MouseX, MouseY);

**delay**(100);

**log\_screen**();

**mouseinit**();

    while (page == 3)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (username.flag != 1 && password.flag != 1)

        {

            while (**bioskey**(1))

            {

**getch**();

            }

        }

**input\_info**(96 + 128 + 4, 128 + 4, 7, 1, &username);

**input\_info**(96 + 128 + 4, 188 + 4, 7, 0, &password);

        if (**mouse\_press**(608, 0, 640, 32) == 2) *//退出框且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

            }

        }

        else if (**mouse\_press**(608, 0, 640, 32) == 1) *//退出框且点击*

        {

            page = 0;

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 2) *//返回框且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**rectangle**(0, 0, 32, 32);

            }

        }

        else if (**mouse\_press**(0, 0, 32, 32) == 1) *//返回框且点击*

        {

            page = 1;

        }

        else if (**mouse\_press**(262, 258, 262 + 120, 258 + 40) == 2) *//登录按钮且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**rectangle**(262, 258, 262 + 120, 258 + 40);

            }

        }

        else if (**mouse\_press**(262, 258, 262 + 120, 258 + 40) == 1) *//登录按钮且点击*

        {

*//清除提示信息*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(96 + 448 - 16 \* 6, 128 + 32 + 4, 96 + 448, 128 + 32 + 4 + 16);

**bar**(96 + 448 - 16 \* 5, 188 + 32 + 4, 96 + 448, 188 + 32 + 4 + 16);

*//确认是否输入完全*

            if (username.str[0] == '\0')

            {

**puthz**(96 + 448 - 16 \* 6, 128 + 32 + 4, "未输入用户名", 16, 16, RED);

            }

            else if (password.str[0] == '\0')

            {

**puthz**(98 + 488 - 16 \* 5, 188 + 32 + 4, "未输入密码", 16, 16, RED);

            }

*//判断用户名是否存在*

            else if (!**judge\_samename**(username.str))

            {

**puthz**(96 + 448 - 16 \* 6, 128 + 32 + 4, "用户名不存在", 16, 16, RED);

            }

*//判断密码是否正确*

            else if (!**right\_key**(username.str, password.str))

            {

**puthz**(96 + 448 - 16 \* 4, 188 + 32 + 4, "密码错误", 16, 16, RED);

            }

*//登陆成功，用参数保存用户名，进入到编辑主页面*

            else

            {

**strcpy**(account\_name, username.str);

                page = 5;

            }

        }

        else if (**mouse\_press**(430, 328, 430 + 128, 328 + 40) == 2) *//忘记密码按钮且未点击*

        {

            pre\_state = state;

            state = 4;

            if (pre\_state != 4) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**rectangle**(430, 328, 430 + 128, 328 + 40);

            }

        }

        else if (**mouse\_press**(430, 328, 430 + 128, 328 + 40) == 1) *//忘记密码按钮且点击*

        {

            page = 4;

        }

        else if (**mouse\_press**(96 + 128, 128, 96 + 448, 128 + 32) == 2 || **mouse\_press**(96 + 128, 188, 96 + 448, 188 + 32) == 2) *//两个输入白框且未点击 还未制作点击状态*

        {

            pre\_state = state;

            state = 5;

            if (pre\_state != 5) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 2; *//光标*

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**rectangle**(0, 0, 32, 32);

                break;

            case 3:

                MouseS = 0;

**setcolor**(LIGHTGRAY);

**setlinestyle**(0, 0, 3);

**rectangle**(262, 258, 262 + 120, 258 + 40);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(262, 258, 262 + 120, 258 + 40);

                break;

            case 4:

                MouseS = 0;

**setcolor**(LIGHTGRAY);

**setlinestyle**(0, 0, 3);

**rectangle**(430, 328, 430 + 128, 328 + 40);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(430, 328, 430 + 128, 328 + 40);

                break;

            case 5:

                MouseS = 0;

                break;

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: log\_screen*

*Function：绘制登陆页面*

*Calls: 无*

*Called By: log*

*Parameter: 无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **log\_screen**(void)

{

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//退出程序的按钮*

**setlinestyle**(0, 0, 1);

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(608, 0, 640, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(608, 0, 640, 32);

**setlinestyle**(0, 0, 3);

**line**(616, 8, 632, 24);

**line**(616, 24, 632, 8);

*//返回按钮*

**setlinestyle**(0, 0, 1);

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(0, 0, 32, 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(0, 0, 32, 32);

**setlinestyle**(0, 0, 3);

**line**(24, 4, 8, 16);

**line**(8, 16, 24, 28);

*//背景*

**setlinestyle**(0, 0, 3);

**setcolor**(WHITE);

**rectangle**(56, 64, 582, 416);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(56, 64, 582, 416);

*//用户名框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 128, 96 + 128, 128 + 32);

**puthz**(96 + 4, 128 + 4, "用户名", 24, 48, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 128, 96 + 448, 128 + 32);

*//密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(96, 188, 96 + 128, 188 + 32);

**puthz**(96 + 20, 188 + 4, "密码", 24, 64, WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(96 + 128, 188, 96 + 448, 188 + 32);

*//登录框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**setlinestyle**(0, 0, 1);

**setcolor**(WHITE);

**bar**(262, 258, 262 + 120, 258 + 40);

**rectangle**(262, 258, 262 + 120, 258 + 40);

**puthz**(262 + 20, 258 + 8, "登录", 24, 56, WHITE);

*//忘记密码框*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**setlinestyle**(0, 0, 1);

**setcolor**(WHITE);

**bar**(430, 328, 430 + 128, 328 + 40);

**rectangle**(430, 328, 430 + 128, 328 + 40);

**puthz**(430 + 4, 328 + 8, "忘记密码", 24, 32, LIGHTRED);

}

9.5 Help.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  帮助界面*

*Function list :  help   帮助界面功能*

*help\_screen  绘制帮助界面*

*help\_page1 第一页*

*help\_page2 第二页*

*help\_page3 第三页*

*help\_page4 第四页*

*Attention:*

*Author:  贾田旺*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "help.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help*

*Function：完成帮助界面功能*

*Calls: help\_screen*

*Called By: main*

*Parameter: 无*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **help**(void)

{

    int index = 1; *//表示翻页索引 目前范围1~4*

    int click = 0; *//表示方向键是否曾有效点击 0未点击 1点击*

    int page = 6;

    int a[2][8] = {{640 - 8 - 4,

                    16,

                    640 - 8 - 4 - 22,

                    4,

                    640 - 8 - 4 - 22,

                    28,

                    640 - 8 - 4,

                    16},

                   {640 - 8 - 30 - 8 - 4 - 22,

                    16,

                    640 - 8 - 30 - 8 - 4,

                    4,

                    640 - 8 - 30 - 8 - 4,

                    28,

                    640 - 8 - 30 - 8 - 4 - 22,

                    16}};         *//绘制方向键数组 右，左*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态 1 返回键*

**clrmous**(MouseX, MouseY);

**delay**(100);

**help\_page1**(a);

**mouseinit**();

*//newmouse(&MouseX, &MouseY, &press); 可以用newmouse不会跳鼠标*

    while (page == 6)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (**mouse\_press**(8, 2, 8 + 48 + 8, 30) == 2) *//返回键且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(8, 2, 8 + 48 + 8, 30);

            }

        }

        else if (**mouse\_press**(8, 2, 8 + 48 + 8, 30) == 1)

        {

            page = 1;

        }

        else if (**mouse\_press**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30) == 2) *//方向左未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**rectangle**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30);

**setfillstyle**(SOLID\_FILL, LIGHTRED);

**fillpoly**(4, a[1]);

            }

        }

        else if (**mouse\_press**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30) == 1 && index > 1) *//方向键左且点击*

        {

            click = 1;

            index -= 1;

**clrmous**(MouseX, MouseY);

**delay**(100);

        }

        else if (**mouse\_press**(640 - 8 - 30, 2, 640 - 8, 30) == 2) *//方向右未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**rectangle**(640 - 8 - 30, 2, 640 - 8, 30);

**setfillstyle**(SOLID\_FILL, LIGHTRED);

**fillpoly**(4, a[0]);

            }

        }

        else if (**mouse\_press**(640 - 8 - 30, 2, 640 - 8, 30) == 1 && index < 4) *//方向键右且点击*

        {

            click = 1;

            index += 1;

**clrmous**(MouseX, MouseY);

**delay**(100);

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(8, 2, 8 + 48 + 8, 30);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**rectangle**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30);

**setfillstyle**(SOLID\_FILL, WHITE);

**fillpoly**(4, a[1]);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**rectangle**(640 - 8 - 30, 2, 640 - 8, 30);

**setfillstyle**(SOLID\_FILL, WHITE);

**fillpoly**(4, a[0]);

                break;

            }

        }

        if (click == 1)

        {

            click = 0;

            switch (index)

            {

            case 1:

**help\_page1**(a);

                break;

            case 2:

**help\_page2**(a);

                break;

            case 3:

**help\_page3**(a);

                break;

            case 4:

**help\_page4**(a);

                break;

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help\_screen*

*Function：绘制帮助界面*

*Calls: 无*

*Called By: help*

*Parameter: a 绘制方向键数组*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **help\_screen**(int (\*a)[8])

{

**cleardevice**();

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 0, 640, 480); *//背景*

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 32);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(640 - 8 - 30, 2, 640 - 8, 30);

**bar**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30);

**bar**(8, 2, 8 + 48 + 8, 30); *//返回键*

**setcolor**(WHITE);

**setfillstyle**(SOLID\_FILL, WHITE);

**fillpoly**(4, a[0]);

**fillpoly**(4, a[1]); *//方向键*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(8, 2, 8 + 48 + 8, 30);

**rectangle**(640 - 8 - 30, 2, 640 - 8, 30);

**rectangle**(640 - 8 - 30 - 8 - 30, 2, 640 - 8 - 30 - 8, 30);

**puthz**(297, 4, "帮助", 24, 32, BLACK);

**puthz**(8 + 4, 4, "返回", 24, 24, WHITE);

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help\_page2*

*Function：帮助页面第一页*

*Calls: 无*

*Called By: help*

*Parameter: a 绘制方向键数组*

*Return: 无*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **help\_page1**(int (\*a)[8])

{

**cleardevice**();

**help\_screen**(a);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(1, HORIZ\_DIR, 2);

**setcolor**(BLACK);

**outtextxy**(432, 0, "Page 1");

**puthz**(0, 46, "用户可在欢迎界面进行以下操作", 16, 16, BLACK); *//从46开始间隔为6*

**puthz**(14 \* 16, 46 + 22 \* 1, "注册：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 1, "需要用户名，身份证号，密码（最少八位）", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 2, "登录：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 2, "使用已注册的用户名和密码进行登录", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 3, "修改密码：", 16, 16, RED);

**puthz**(19 \* 16, 46 + 22 \* 3, "使用已注册的用户名与对应身份证号修改密码", 16, 16, BLACK);

**puthz**(0, 46 + 22 \* 6, "登录", 16, 16, RED);

**puthz**(16 \* 2, 46 + 22 \* 6, "后进入导图操作页面", 16, 16, BLACK);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(0, 46 + 22 \* 6, 16 \* 2, 46 + 22 \* 6 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 1, 14 \* 16 + 16 \* 2, 46 + 22 \* 1 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 2, 14 \* 16 + 16 \* 2, 46 + 22 \* 2 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 3, 14 \* 16 + 16 \* 4, 46 + 22 \* 3 + 16 \* 1);

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help\_page2*

*Function：帮助页面第二页*

*Calls: 无*

*Called By: help*

*Parameter: a 绘制方向键数组*

*Return: 无*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **help\_page2**(int (\*a)[8])

{

**cleardevice**();

**help\_screen**(a);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(1, HORIZ\_DIR, 2);

**setcolor**(BLACK);

**outtextxy**(432, 0, "Page 2");

**puthz**(0, 46, "用户可在操作界面进行以下操作", 16, 16, BLACK); *//从46开始间隔为6*

**puthz**(14 \* 16, 46 + 22 \* 1, "新建：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 1, "新建一个导图（名字不可重复），可选类型有四种", 16, 16, BLACK);

**puthz**(17 \* 16, 46 + 22 \* 2, "单向导图，树状图，时间轴，鱼骨图", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 3, "打开：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 3, "输入名称可打开已存在导图文件", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 4, "关闭：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 4, "关闭当前导图文件，并显示初始页面", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 5, "删除：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 5, "关闭当前导图文件，显示初始页面并删除对应导图", 16, 16, BLACK);

**puthz**(17 \* 16, 46 + 22 \* 6, "文件", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 7, "编辑：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 7, "（已打开导图文件）对导图框内容进行编辑子操作", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 8, "保存：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 8, "对更改进行保存", 16, 16, BLACK);

**puthz**(0, 46 + 22 \* 11, "点击", 16, 16, BLACK);

**puthz**(2 \* 16, 46 + 22 \* 11, "编辑", 16, 16, RED);

**puthz**(4 \* 16, 46 + 22 \* 11, "后进入编辑界面", 16, 16, BLACK);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(14 \* 16, 46 + 22 \* 1, 14 \* 16 + 2 \* 16, 46 + 22 \* 1 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 5, 14 \* 16 + 2 \* 16, 46 + 22 \* 5 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 3, 14 \* 16 + 2 \* 16, 46 + 22 \* 3 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 4, 14 \* 16 + 2 \* 16, 46 + 22 \* 4 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 7, 14 \* 16 + 2 \* 16, 46 + 22 \* 7 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 8, 14 \* 16 + 2 \* 16, 46 + 22 \* 8 + 16 \* 1);

**rectangle**(2 \* 16, 46 + 22 \* 11, 2 \* 16 + 2 \* 16, 46 + 22 \* 11 + 16 \* 1);

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help\_page3*

*Function：帮助页面第三页*

*Calls: 无*

*Called By: help*

*Parameter: a 绘制方向键数组*

*Return: 无*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **help\_page3**(int (\*a)[8])

{

**cleardevice**();

**help\_screen**(a);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(1, HORIZ\_DIR, 2);

**setcolor**(BLACK);

**outtextxy**(432, 0, "Page 3");

**puthz**(0, 46, "用户可在单向思维导图、树状图、鱼骨图编辑界面进行以下操作：", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 1, "选中：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 1, "点击存在的思维导图框进行选中，标红", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 2, "编辑文本：", 16, 16, RED);

**puthz**(19 \* 16, 46 + 22 \* 2, "标红框后立刻进入文本编辑状态，按下", 16, 16, BLACK);

**puthz**(19 \* 16 + 17 \* 16, 46 + 22 \* 2, "回车", 16, 16, RED);

**puthz**(21 \* 16 + 17 \* 16, 46 + 22 \* 2, "退", 16, 16, BLACK);

**puthz**(19 \* 16, 46 + 22 \* 3, "出状态", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 4, "新建：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 4, "标红后新建子框", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 5, "删除：", 16, 16, RED);

**puthz**(17 \* 16, 46 + 22 \* 5, "删除标红框及子框，界面调整", 16, 16, BLACK);

**puthz**(0, 46 + 22 \* 8, "单向思维导图、树状图可改变框色，字色：", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 9, "选中框后进入标红状态，点击颜色，选择", 16, 16, BLACK);

**puthz**(14 \* 16 + 18 \* 16, 46 + 22 \* 9, "框色", 16, 16, RED);

**puthz**(14 \* 16 + 20 \* 16, 46 + 22 \* 9, "或", 16, 16, BLACK);

**puthz**(14 \* 16 + 21 \* 16, 46 + 22 \* 9, "字色", 16, 16, RED);

**puthz**(14 \* 16 + 23 \* 16, 46 + 22 \* 9, "即可", 16, 16, BLACK);

**puthz**(0, 46 + 22 \* 10, "用户在时间轴编辑界面可进行以下操作：", 16, 16, BLACK);

**setcolor**(RED);

**puthz**(14 \* 16, 46 + 22 \* 11, "输入正确格式年月日时间及内容点击新增，", 16, 16, BLACK); *//蓝绿红黑*

**puthz**(14 \* 16 + 19 \* 16, 46 + 22 \* 11, "新增", 16, 16, RED);

**puthz**(14 \* 16 + 21 \* 16, 46 + 22 \* 11, "时间轴框", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 12, "选中导图框进入标红状态，点击", 16, 16, BLACK);

**puthz**(14 \* 16 + 14 \* 16, 46 + 22 \* 12, "删除", 16, 16, RED);

**puthz**(14 \* 16 + 16 \* 16, 46 + 22 \* 12, "对应框", 16, 16, BLACK);

**puthz**(14 \* 16, 46 + 22 \* 13, "导图中", 16, 16, BLACK);

**puthz**(14 \* 16 + 3 \* 16, 46 + 22 \* 13, "蓝色", 16, 16, BLUE);

**puthz**(14 \* 16 + 5 \* 16, 46 + 22 \* 13, "对应年份，", 16, 16, BLACK);

**puthz**(14 \* 16 + 10 \* 16, 46 + 22 \* 13, "绿色", 16, 16, GREEN);

**puthz**(14 \* 16 + 12 \* 16, 46 + 22 \* 13, "对应月份，", 16, 16, BLACK);

**puthz**(14 \* 16 + 17 \* 16, 46 + 22 \* 13, "红色", 16, 16, RED);

**puthz**(14 \* 16 + 19 \* 16, 46 + 22 \* 13, "对应天", 16, 16, BLACK);

**puthz**(0, 46 + 22 \* 16, "点击", 16, 16, BLACK);

**puthz**(2 \* 16, 46 + 22 \* 16, "完成", 16, 16, RED);

**puthz**(4 \* 16, 46 + 22 \* 16, "后退出编辑页面", 16, 16, BLACK);

**setlinestyle**(0, 0, 1);

**rectangle**(19 \* 16 + 17 \* 16, 46 + 22 \* 2, 38 \* 16, 46 + 22 \* 2 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 4, 14 \* 16 + 2 \* 16, 46 + 22 \* 4 + 16 \* 1);

**rectangle**(14 \* 16, 46 + 22 \* 5, 14 \* 16 + 2 \* 16, 46 + 22 \* 5 + 16 \* 1);

**rectangle**(2 \* 16, 46 + 22 \* 16, 2 \* 16 + 2 \* 16, 46 + 22 \* 16 + 16 \* 1);

**rectangle**(14 \* 16 + 18 \* 16, 46 + 22 \* 9, 14 \* 16 + 20 \* 16, 46 + 22 \* 9 + 16);

**rectangle**(14 \* 16 + 21 \* 16, 46 + 22 \* 9, 14 \* 16 + 23 \* 16, 46 + 22 \* 9 + 16);

**rectangle**(14 \* 16 + 19 \* 16, 46 + 22 \* 11, 14 \* 16 + 21 \* 16, 46 + 22 \* 11 + 16);

**rectangle**(14 \* 16 + 14 \* 16, 46 + 22 \* 12, 14 \* 16 + 16 \* 16, 46 + 22 \* 12 + 16);

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: help\_page4*

*Function：帮助页面第四页*

*Calls: 无*

*Called By: help*

*Parameter: a 绘制方向键数组*

*Return: 无*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **help\_page4**(int (\*a)[8])

{

**cleardevice**();

**help\_screen**(a);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(1, HORIZ\_DIR, 2);

**setcolor**(BLACK);

**outtextxy**(432, 0, "Page 4");

**puthz**(0, 46, "新建导图文件界面", 16, 16, BLACK);

**puthz**(8 \* 16, 46 + 1 \* 16, "用户可选中导图类型，输入导图名进行新建", 16, 16, BLACK);

**puthz**(8 \* 16, 46 + 2 \* 16, "也可退出程序，返回操作界面", 16, 16, BLACK);

**puthz**(0, 46 + 4 \* 16, "打开导图文件界面", 16, 16, BLACK);

**puthz**(8 \* 16, 46 + 5 \* 16, "输入导图名，点击确定打开", 16, 16, BLACK);

**puthz**(8 \* 16, 46 + 6 \* 16, "点击取消，返回操作界面", 16, 16, BLACK);

}

9.6 EnterT.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description: 登录注册所需工具函数，操作界面所需部分工具函数*

*Function list :  input\_info 键盘输入显示函数*

*judge\_samename 检查用户名是否已存在函数*

*judge\_ID 检查身份证号格式是否符合要求*

*judge\_password 检查密码是否符合要求*

*write\_info 将账户用户名，身份证号，密码写入账户文件*

*new\_file 为新注册的用户建立一个文件保存其账户信息*

*right\_ID 判断身份证号是否正确*

*rewrite\_key 写入新密码*

*right\_key 判断密码是否正确*

*findmind 根据传入的用户名与思维导图名找到用户中的MIND结构体*

*delete\_mind 删除用户文件中的指定MIND结构体,并删除对应的导图文件*

*Attention:*

*Author： 贾田旺*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "EnterT.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: input\_info*

*Function：键盘输入显示函数*

*Calls: 无*

*Called By: Signup*

*Log*

*ResetKey*

*Parameter: x  输入位置左边界*

*y  输入位置上边界*

*size  字符大小*

*mode  输入模式，1为可见信息如用户名、身份证号，0为不可见信息如密码*

*content  输入信息的结构体，详见data.h*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **input\_info**(int x, int y, int size, int mode, ENTER \*content) *//将信息输入到content中，并显示在屏幕上*

{

    char tmp;

    int i;

    if (press == 1)

    {

        if (**mouse\_press**(content->x1, content->y1, content->x2, content->y2) == 1) *//在输入框内点击*

        {

            content->flag = 1;

**clrmous**(MouseX, MouseY);

**delay**(5); *//FIXME*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(content->x1, content->y1, content->x2, content->y2);

**setcolor**(DARKGRAY);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

        }

        else *//在输入框外点击*

        {

            content->flag = 0;

**clrmous**(MouseX, MouseY);

**delay**(5); *//FIXME*

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(content->x1, content->y1, content->x2, content->y2);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

        }

    }

    if (content->flag == 1) *//flag为1时表示可以接收键盘输入*

    {

        if (**kbhit**())

        {

            tmp = **getch**();

            if (tmp >= '!' && tmp <= '~') *//输入一个字符*

            {

                if (content->cursor < content->length) *//不能超过最大长度*

                {

                    (content->str)[content->cursor] = tmp;

                    (content->str)[content->cursor + 1] = '\0';

                    (content->cursor)++;

                }

            }

            else if (tmp == '\b') *//删除字符*

            {

                if (content->cursor > 0)

                {

                    (content->str)[content->cursor - 1] = '\0';

                    (content->cursor)--;

                }

            }

**setcolor**(DARKGRAY);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, WHITE);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, size);

**bar**(content->x1 + 2, content->y1 + 2, content->x2 - 2, content->y2 - 2); *//每一步显示之前先用背景色覆盖掉原来信息*

            if (mode == 1) *//显示字符串并划线*

            {

**setcolor**(DARKGRAY);

**clrmous**(MouseX, MouseY);

**delay**(5); *//FIXME*

**outtextxy**(x, y, content->str);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

            }

            else *//显示\**

            {

**setcolor**(DARKGRAY);

                for (i = 0; i < content->cursor; i++)

                {

**clrmous**(MouseX, MouseY);

**delay**(5); *//FIXME*

**outtextxy**(x + i \* (2 \* size - 2), y, "\*");

                }

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: judge\_samename*

*Function：检查用户名是否已存在*

*Calls: 无*

*Called By: Signup*

*Log*

*ResetKey*

*Parameter: name 用户名*

*Return: flag  0表示不存在，1表示存在*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **judge\_samename**(char \*name)

{

    FILE \*fp;

    USER \*user = NULL;

    int flag = 0; *//返回的是否存在标志*

    int num;      *//文件用户个数*

    int i;

    if ((fp = **fopen**(".\\UserInfo\\UserInfo.dat", "rb")) == NULL) *//打开失败，异常退出*

    {

**closegraph**();

**printf**("\nFile UserInfo opening error.(from judge\_samename)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0, SEEK\_END);

    num = **ftell**(fp) / sizeof(USER); *//得到用户数量*

    if (num == 0)

    {

        flag = 0;

        return flag;

    }

    user = (USER \*)**malloc**(sizeof(USER));

    if (user == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from juse\_samename)\n");

**delay**(3000);

**exit**(1);

    }

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(USER), SEEK\_SET);

**fread**(user, sizeof(USER), 1, fp);

        if (!**strcmp**(user->username, name))

        {

            flag = 1;

            break;

        }

    }

**free**(user);

    user = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from judge\_samename)\n");

**delay**(3000);

**exit**(1);

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: judge\_ID*

*Function：检查身份证号是否符合要求*

*Calls: 无*

*Called By: Signup*

*ResetKey*

*Parameter: card 身份证号*

*Return: flag  0表示错误，1表示正确*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **judge\_ID**(char \*card)

{

    int i;

    int flag = 1; *//表示身份证号是否符合要求*

    if (**strlen**(card) != 18)

    {

        flag = 0;

        return flag;

    }

    for (i = 0; i < 18; i++)

    {

        if ((i < 17) && (card[i] < '0' || card[i] > '9')) *//若在除最后一位之外出现非数字字符，则错误*

        {

            flag = 0;

            break;

        }

        if ((i == 17) && (card[i] < '0' || card[i] > '9') && (card[i] != 'x' || card[i] != 'X')) *//若在最后一位出现不是x或X的非数字字符，则错误*

        {

            flag = 0;

            break;

        }

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: judge\_password*

*Function：判断密码是否符合要求*

*Calls: 无*

*Called By: Signup*

*ResetKey*

*Parameter: password 密码*

*Return: 1 符合要求  0 不符合要求*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **judge\_password**(char \*password)

{

    if (**strlen**(password) < 8) *//密码小于8位不符合要求*

        return 0;

    else

        return 1;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: write\_info*

*Function：将账户用户名，身份证号，密码写入账户文件*

*Calls: 无*

*Called By: Signup*

*ResetKey*

*Parameter: name 用户名*

*card 身份证号*

*password 密码*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **write\_info**(char \*name, char \*card, char \*password)

{

    FILE \*fp;

    USER \*user;

    int i;

    if ((fp = **fopen**(".\\UserInfo\\UserInfo.dat", "ab+")) == NULL) *//以读写追加方式打开*

    {

*//打开失败，异常退出*

**closegraph**();

**printf**("\nFile UserInfo opening error.(from write\_info)\n");

**delay**(3000);

**exit**(1);

    }

    user = (USER \*)**malloc**(sizeof(USER));

    if (user == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from write\_info)\n");

**delay**(3000);

**exit**(1);

    }

**strcpy**(user->username, name);

**strcpy**(user->ID, card);

**strcpy**(user->password, password);

**fwrite**(user, sizeof(USER), 1, fp);

**free**(user);

    user = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from write\_info)\n");

**delay**(3000);

**exit**(1);

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: new\_file*

*Function：为新注册的用户建立一个文件保存其账户信息*

*Calls: 无*

*Called By: Signup*

*Parameter: name 新用户姓名*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **new\_file**(char \*name)

{

    FILE \*fp;

    char filename[25 + 1]; *//新用户文件名*

**sprintf**(filename, ".\\UserInfo\\%s.dat", name);

    if ((fp = **fopen**(filename, "wb+")) == NULL) *//为读写建立一个新的二进制文件。*

    {

*//打开失败，异常退出*

**closegraph**();

**printf**("\nFile opening error.(from new\_file)\n");

**delay**(3000);

**exit**(1);

    }

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile closed error.(from new\_file)\n");

**delay**(3000);

**exit**(1);

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: right\_ID*

*Function：判断身份证号是否正确*

*Calls: 无*

*Called By: ResetKey*

*Parameter: name 用户名*

*card 用户输入的身份证号*

*Return: flag  1 表示正确  2 表示错误*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **right\_ID**(char \*name, char \*card)

{

    FILE \*fp;

    USER \*user;

    int flag = 1;     *//表示身份证号是否正确*

    int i, num, j, k; *//文件用户个数*

    if ((fp = **fopen**(".\\UserInfo\\UserInfo.dat", "rb")) == NULL) *//以只读方式打开*

    {

*//打开失败，退出程序*

**closegraph**();

**printf**("\nFile UserInfo opening error.(from rihgt\_ID)\n");

**delay**(3000);

**exit**(1);

    }

    user = (USER \*)**malloc**(sizeof(USER));

    if (user == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from right\_ID)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0, SEEK\_END);

    num = **ftell**(fp) / sizeof(USER); *//得到用户数量*

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(USER), SEEK\_SET);

**fread**(user, sizeof(USER), 1, fp);

        if (**strcmp**(user->username, name) == 0)

        {

            break;

        }

    }

    if (**strcmp**(user->ID, card) == 0) *//如果输入身份证号与文件内部相同*

    {

        flag = 1;

    }

    else if (**strcmp**(user->ID, card) != 0)

    {

        flag = 0;

    }

**free**(user);

    user = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from right\_ID)\n");

**delay**(3000);

**exit**(1);

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: rewrite\_key*

*Function：写入新密码*

*Calls: 无*

*Called By: ResetKey*

*Parameter: name 用户名*

*newkey 新密码*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **rewrite\_key**(char \*name, char \*newkey)

{

    FILE \*fp;

    USER \*user;

    int i, num; *//文件用户个数*

    if ((fp = **fopen**(".\\UserInfo\\UserInfo.dat", "rb+")) == NULL) *//以读写方式打开*

    {

*//打开失败，退出程序*

**closegraph**();

**printf**("\nFile UserInfo opening error.(from rewrite\_key)\n");

**delay**(3000);

**exit**(1);

    }

    user = (USER \*)**malloc**(sizeof(USER));

    if (user == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from rewrite\_key)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0, SEEK\_END);

    num = **ftell**(fp) / sizeof(USER); *//得到用户数量*

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(USER), SEEK\_SET);

**fread**(user, sizeof(USER), 1, fp);

        if (!**strcmp**(user->username, name))

        {

            break;

        }

    }

**strcpy**(user->password, newkey);

**fseek**(fp, i \* sizeof(USER), SEEK\_SET);

**fwrite**(user, sizeof(USER), 1, fp);

**free**(user);

    user = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from rewite\_key)\n");

**delay**(3000);

**exit**(1);

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: right\_key*

*Function：判断密码是否正确*

*Calls: 无*

*Called By: Log*

*Parameter: name 用户名*

*password  用户输入密码*

*Return: flag 1 表示正确 0 表示错误*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **right\_key**(char \*name, char \*password)

{

    FILE \*fp;

    USER \*user;

    int flag = 1;

    int i, num; *//文件用户个数*

    if ((fp = **fopen**(".\\UserInfo\\UserInfo.dat", "rb")) == NULL) *//以只读方式打开*

    {

*//打开失败，退出程序*

**closegraph**();

**printf**("\nFile UserInfo opening error.(from right\_key)\n");

**delay**(3000);

**exit**(1);

    }

    user = (USER \*)**malloc**(sizeof(USER));

    if (user == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from right\_key)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0, SEEK\_END);

    num = **ftell**(fp) / sizeof(USER); *//得到用户数量*

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(USER), SEEK\_SET);

**fread**(user, sizeof(USER), 1, fp);

        if (!**strcmp**(user->username, name))

        {

            break;

        }

    }

    if (**strcmp**(user->password, password) != 0) *//密码不相符*

    {

        flag = 0;

    }

    else if (**strcmp**(user->password, password) == 0) *//密码相符*

    {

        flag = 1;

    }

**free**(user);

    user = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from right\_key)\n");

**delay**(3000);

**exit**(1);

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: findmind*

*Function：根据传入的用户名与思维导图名找到用户中的MIND结构体*

*Calls: 无*

*Called By: Open.c*

*Parameter: account\_name 账户名*

*mindname 思维导图名*

*Return: MIND 结构体*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

MIND **findmind**(char \*account\_name, char \*mindname)

{

    FILE \*fp;

    MIND mind1;

    int num;            *//用户所拥有的导图文件个数*

    int i;              *//计数用*

    char place[25 + 1]; *//存放用户所拥有的文件的信息索引*

**sprintf**(place, ".\\UserInfo\\%s.dat", account\_name);

    if ((fp = **fopen**(place, "rb")) == NULL) *//打开失败，异常退出*

    {

**closegraph**();

**printf**("\nFile UserInfo opening error.(from findmind)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0L, SEEK\_END);

    num = **ftell**(fp) / sizeof(MIND); *//得到导图数量*

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(MIND), SEEK\_SET);

**fread**(&mind1, sizeof(MIND), 1, fp);

        if (!**strcmp**(mind1.mindname, mindname))

        {

            break;

        }

    }

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from findmind)\n");

**delay**(3000);

**exit**(1);

    }

    return mind1;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: delete\_mind*

*Function：删除用户文件中的指定MIND结构体,并删除对应的导图文件*

*Calls: 无*

*Called By: edit.c*

*Parameter: account\_name 账户名*

*mindname 导图名*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **delete\_mind**(char \*account\_name, char \*mindname)

{

    FILE \*fp, \*ftmp;

    MIND \*mind1;

    int i, num;                                         *//计数，用户拥有导图个数*

    char place\_user[25 + 1];                            *//存放用户文件的信息索引*

    char place\_tp[24 + 1] = ".\\UserInfo\\tmp\_dmd.dat"; *//暂时*

    char place\_daotu[32 + 1];                           *//导图文件索引*

**sprintf**(place\_user, ".\\UserInfo\\%s.dat", account\_name);

**sprintf**(place\_daotu, ".\\daotu\\%s\\%s.dat", account\_name, mindname);

    if ((fp = **fopen**(place\_user, "rb")) == NULL) *//打开失败，异常退出*

    {

**closegraph**();

**printf**("\nFile UserInfo opening error.(from delete\_mind1)\n");

**delay**(3000);

**exit**(1);

    }

    if ((ftmp = **fopen**(place\_tp, "wb+")) == NULL) *//打开失败，异常退出*

    {

**closegraph**();

**printf**("\nFile UserInfo opening error.(from delete\_mind2)\n");

**delay**(3000);

**exit**(1);

    }

    mind1 = (MIND \*)**malloc**(sizeof(MIND));

    if (mind1 == NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from delete\_mind)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0L, SEEK\_END);

    num = **ftell**(fp) / sizeof(MIND); *//得到导图数量*

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(MIND), SEEK\_SET);

**fread**(mind1, sizeof(MIND), 1, fp);

        if (**strcmp**(mind1->mindname, mindname)) *//如果和要删除的导图MIND不同*

        {

**fwrite**(mind1, sizeof(MIND), 1, ftmp);

        }

    }

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from delete\_mind1)\n");

**delay**(3000);

**exit**(1);

    }

    if (**fclose**(ftmp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from delete\_mind2)\n");

**delay**(3000);

**exit**(1);

    }

**free**(mind1);

    mind1 = NULL;

**remove**(place\_user);

**rename**(place\_tp, place\_user);

**remove**(place\_daotu);

}

9.7 Operate.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  完成操作主页面及相关功能*

*Function list :  operate   操作思维导图功能*

*operate\_screen  绘制操作页面*

*Attention:*

*Author:  贾田旺*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "operate.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: operate*

*Function：编辑思维导图功能*

*Calls:  drawmind*

*drawmind\_tr*

*drawbone*

*drawdate*

*new*

*operate\_screen*

*opena*

*delete\_mind*

*Called By: main*

*Parameter: account\_name*

*Return: page 页面跳转*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **operate**(char \*account\_name)

{

    MIND daotu\_info = {0, ""}; *//导图信息*

    char temdt[30 + 1]; *//当前思维导图文件位置索引*

    FILE \*fp; *//存放当前打开的文件指针*

    int open\_ste; *//存放打开状态*

    MINDBOX oper\_temp[HORI][VERT];

    MINDBOX oper\_temptr[HORI][VERTREE];

    FISHINFO oper\_fish;

    DATEINFO oper\_date;

    int page = 5;

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态 1 退出程序按钮 2 返回欢迎界面按钮*

*//方向键 3 上 4 下 5 左 6 右*

*// 7 新建  8 打开  9 关闭  10 删除*

*// 11 帮助 12 编辑*

**clrmous**(MouseX, MouseY);

**delay**(100);

**operate\_screen**();

**mouseinit**();

    while (page == 5)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (**mouse\_press**(616, 0, 640, 24) == 2) *//退出框且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(616, 0, 640, 24);

**setlinestyle**(0, 0, 3);

**line**(624, 4, 632, 20);

**line**(624, 20, 632, 4);

            }

        }

        else if (**mouse\_press**(616, 0, 640, 24) == 1) *//退出框且点击*

        {

            page = 0;

        }

        else if (**mouse\_press**(496, 0, 496 + 24, 24) == 2) *//返回登录界面框且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496, 0, 496 + 24, 24);

            }

        }

        else if (**mouse\_press**(496, 0, 496 + 24, 24) == 1) *//返回登录界面框且点击*

        {

            page = 1;

        }

        else if (**mouse\_press**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32) == 2) *//新建未点击*

        {

            pre\_state = state;

            state = 7;

            if (pre\_state != 7) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32);

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32) == 1) *//新建点击*

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            if((**new**(account\_name, &daotu\_info) == 1)||daotu\_info.type != 0)

            {

**operate\_screen**();

**puthz**(16, 4, "导图名：", 16, 16, BLACK);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 7);

**setcolor**(BLACK);

**outtextxy**(96, 0, daotu\_info.mindname);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

                if(fp != NULL)

                {

**fclose**(fp);

                }

**sprintf**(temdt, ".\\daotu\\%s\\%s.dat", account\_name,daotu\_info.mindname);*//写在另一个函数里面*

                if((fp = **fopen**(temdt,"rb+")) == NULL)*//打开失败，异常退出*

                {

**closegraph**();

**printf**("\nFile mind opening error.(from new\_in\_edit)\n");

**delay**(3000);

**exit**(1);

                }

                switch(daotu\_info.type)

                {

                    case 1:

**fseek**(fp, 0, 0);

**fread**(&(oper\_temp[0][0]), sizeof(MINDBOX), HORI \* VERT, fp);

**drawmind**(oper\_temp);

                    break;

                    case 2:

**fseek**(fp, 0, 0);

**fread**(&(oper\_temptr[0][0]), sizeof(MINDBOX), HORI \* VERTREE, fp);

**drawmind\_tr**(oper\_temptr);

                    break;

                    case 3:

**fseek**(fp, 0, 0);

**fread**(&(oper\_fish), sizeof(FISHINFO), 1, fp);

**drawbone**(&oper\_fish);

                    break;

                    case 4:

**fseek**(fp, 0, 0);

**fread**(&(oper\_date), sizeof(DATEINFO), 1, fp);

**drawdate**(&oper\_date);

                    break;

                }

            }

            else

            {

**operate\_screen**();

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32) == 2) *//打开且未点击*

        {

            pre\_state = state;

            state = 8;

            if (pre\_state != 8) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32);

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32) == 1) *//打开且点击*

        {

*//opena(account\_name, &daotu\_info);*

**clrmous**(MouseX, MouseY);

**delay**(5);

            open\_ste = **opena**(account\_name, &daotu\_info);

            if (daotu\_info.type != 0)

            {

**operate\_screen**();

**puthz**(16, 4, "导图名：", 16, 16, BLACK);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 7);

**setcolor**(BLACK);

**outtextxy**(96, 0, daotu\_info.mindname);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

            }

            else

            {

**operate\_screen**();

            }

            if(open\_ste == 1)

            {

                if(fp != NULL)

                {

**fclose**(fp);

                }

**sprintf**(temdt, ".\\daotu\\%s\\%s.dat", account\_name, daotu\_info.mindname);

                if((fp = **fopen**(temdt,"rb+")) == NULL)*//打开失败，异常退出*

                {

**closegraph**();

**printf**("\nFile mind opening error.(from open\_in\_edit)\n");

**delay**(3000);

**exit**(1);

                }

            }

            switch(daotu\_info.type)

            {

                case 1:

**fseek**(fp, 0, 0);

**fread**(&(oper\_temp[0][0]), sizeof(MINDBOX), HORI \* VERT, fp);

**drawmind**(oper\_temp);

                break;

                case 2:

**fseek**(fp, 0, 0);

**fread**(&(oper\_temptr[0][0]), sizeof(MINDBOX), HORI \* VERTREE, fp);

**drawmind\_tr**(oper\_temptr);

                break;

                case 3:

**fseek**(fp, 0, 0);

**fread**(&(oper\_fish), sizeof(FISHINFO), 1, fp);

**drawbone**(&oper\_fish);

                break;

                case 4:

**fseek**(fp, 0, 0);

**fread**(&(oper\_date), sizeof(DATEINFO), 1, fp);

**drawdate**(&oper\_date);

                break;

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32) == 2) *//关闭且未点击*

        {

            pre\_state = state;

            state = 9;

            if (pre\_state != 9) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32);

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32) == 1) *//关闭且点击*

        {

            daotu\_info.type = 0;

**strcpy**(daotu\_info.mindname, ""); *//清除信息*

            if(fp != NULL)

            {

**fclose**(fp);

            }

**clrmous**(MouseX, MouseY);

**delay**(5);

**operate\_screen**();

        }

        else if (**mouse\_press**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32) == 2) *//删除且未点击*

        {

            pre\_state = state;

            state = 10;

            if (pre\_state != 10) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32);

            }

        }

        else if (**mouse\_press**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32) == 1) *//删除且点击*

        {

**delete\_mind**(account\_name, daotu\_info.mindname); *//删除操作*

            daotu\_info.type = 0;

**strcpy**(daotu\_info.mindname, ""); *//清除MIND结构体信息*

**clrmous**(MouseX, MouseY);

**delay**(5);

**operate\_screen**();

        }

        else if (**mouse\_press**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32) == 2) *//编辑未点击*

        {

            pre\_state = state;

            state = 12;

            if(pre\_state != 12)*//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32);

            }

        }

        else if(**mouse\_press**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32) == 1) *//编辑且点击*

        {

            if(fp != NULL)

            {

                switch(daotu\_info.type)

                {

                    case 1:

**edit**(oper\_temp);

                    break;

                    case 2:

**edittr**(oper\_temptr);

                    break;

                    case 3:

**editfb**(&oper\_fish);

                    break;

                    case 4:

**edidate**(&oper\_date);

                    break;

                }

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

**operate\_screen**();

                if(daotu\_info.type != 0)

                {

**puthz**(16, 4, "导图名：", 16, 16, BLACK);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 7);

**setcolor**(BLACK);

**outtextxy**(96, 0, daotu\_info.mindname);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

                }

                switch(daotu\_info.type)

                {

                    case 1:

**drawmind**(oper\_temp);

                    break;

                    case 2:

**drawmind\_tr**(oper\_temptr);

                    break;

                    case 3:

**drawbone**(&oper\_fish);

                    break;

                    case 4:

**drawdate**(&oper\_date);

                    break;

                }

            }

        }

        else if(**mouse\_press**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32) == 2) *//保存未点击*

        {

            pre\_state = state;

            state = 13;

            if(pre\_state != 13)*//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32);

            }

        }

        else if(**mouse\_press**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32) == 1) *//保存已点击*

        {

            if(fp != NULL)

            {

                switch(daotu\_info.type)

                {

                    case 1:

**fseek**(fp, 0, 0);

**fwrite**(&(oper\_temp[0][0]), sizeof(MINDBOX), HORI \* VERT, fp);

                    break;

                    case 2:

**fseek**(fp, 0, 0);

**fwrite**(&(oper\_temptr[0][0]), sizeof(MINDBOX), HORI \* VERTREE, fp);

                    break;

                    case 3:

**fseek**(fp, 0, 0);

**fwrite**(&oper\_fish, sizeof(FISHINFO), 1, fp);

                    break;

                    case 4:

**fseek**(fp, 0, 0);

**fwrite**(&oper\_date, sizeof(DATEINFO), 1, fp);

                    break;

                }

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(616, 0, 640, 24);

**setlinestyle**(0, 0, 3);

**line**(624, 4, 632, 20);

**line**(624, 20, 632, 4);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496, 0, 496 + 24, 24);

                break;

            case 7:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32);

                break;

            case 8:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32);

                break;

            case 9:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32);

                break;

            case 10:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32);

                break;

            case 11:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 20, 0, 496 + 24 + 20 + 56, 0 + 24);

                break;

            case 12:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32);

            case 13:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32);

            }

        }

    }

    return page;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: operate\_screen*

*Function：绘制操作页面*

*Calls: 无*

*Called By: operate*

*Parameter: 无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **operate\_screen**(void)

{

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 0, 496, 480);

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(496, 0, 640, 480);

**bar**(0, 0, 496, 24);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**line**(496, 0, 496, 480);

*//退出程序的按钮*

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(616, 0, 640, 24);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(616, 0, 640, 24);

**setlinestyle**(0, 0, 3);

**line**(624, 4, 632, 20);

**line**(624, 20, 632, 4);

*//退出登录的按钮*

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(496, 0, 496 + 24, 24);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496, 0, 496 + 24, 24);

**setlinestyle**(0, 0, 3);

**line**(496 + 16, 4, 496 + 8, 12);

**line**(496 + 8, 12, 496 + 16, 20);

*//功能按钮*

**setcolor**(LIGHTGRAY);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32);

**bar**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32);

**bar**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32);

**bar**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32);

**bar**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32);

**bar**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 24 + 12, 40, 496 + 24 + 12 + 72, 40 + 32);

**rectangle**(496 + 24 + 12, 72 + 8, 496 + 24 + 12 + 72, 72 + 8 + 32);

**rectangle**(496 + 24 + 12, 112 + 8, 496 + 24 + 12 + 72, 112 + 8 + 32);

**rectangle**(496 + 24 + 12, 152 + 8, 496 + 24 + 12 + 72, 152 + 8 + 32);

**rectangle**(496 + 24 + 12, 192 + 8, 496 + 24 + 12 + 72, 192 + 8 + 32);

**rectangle**(496 + 24 + 12, 232 + 8, 496 + 24 + 12 + 72, 232 + 8 + 32);

**puthz**(496 + 24 + 12 + 8, 40 + 4, "新建", 24, 32, WHITE);

**puthz**(496 + 24 + 12 + 8, 72 + 8 + 4, "打开", 24, 32, WHITE);

**puthz**(496 + 24 + 12 + 8, 112 + 8 + 4, "关闭", 24, 32, WHITE);

**puthz**(496 + 24 + 12 + 8, 152 + 8 + 4, "删除", 24, 32, WHITE);

**puthz**(496 + 24 + 12 + 8, 192 + 8 + 4, "编辑", 24, 32, WHITE);

**puthz**(496 + 24 + 12 + 8, 232 + 8 + 4, "保存", 24, 32, WHITE);

*//底字*

**puthz**(152, 216, "欢迎来到思维导图", 24, 24, LIGHTGRAY);

}

9.8 OperateT.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑页面及附属页面所用工具函数*

*Function list :  daotujud  判断导图文件是否重复*

*newmind   新建一个导图文件*

*input\_mind 输入并显示信息函数*

*newdrt 新建一个存放新用户导图文件的文件夹*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "operateT.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: daotujud*

*Function：判断导图文件是否重复*

*Calls:无*

*Called By: new Open.c*

*Parameter：account\_name 用户名*

*mind\_name 待新建的思维导图文件名*

*Return: 标识，1为重复，0未重复*

*Author: 倪启源*

*Others:无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **daotujud**(char \*account\_name, char \*mind\_name)

{

    FILE \*fp;

    MIND \*mind = NULL;

    int flag = 0;       *//返回导图文件是否存在标志,1存在，0不存在*

    int num;            *//用户所拥有的导图文件个数*

    int i;              *//计数用*

    char place[25 + 1]; *//存放用户所拥有的文件的信息索引*

**sprintf**(place, ".\\UserInfo\\%s.dat", account\_name);

    if ((fp = **fopen**(place, "rb")) == NULL) *//打开失败，异常退出*

    {

**closegraph**();

**printf**("\nFile UserInfo opening error.(from daotujud\_open)\n");

**delay**(3000);

**exit**(1);

    }

**fseek**(fp, 0L, SEEK\_END);

    num = **ftell**(fp) / sizeof(MIND); *//得到导图数量*

    if (num == 0)                   *//无导图*

    {

        flag = 0;

        return flag;

    }

    mind = (MIND \*)**malloc**(sizeof(MIND));

    if (mind== NULL) *//分配内存失败，退出程序*

    {

**closegraph**();

**printf**("\nMemory location failed.(from daotujud\_malloc)\n");

**delay**(3000);

**exit**(1);

    }

    for (i = 0; i < num; i++)

    {

**fseek**(fp, i \* sizeof(MIND), SEEK\_SET);

**fread**(mind, sizeof(MIND), 1, fp);

        if (!**strcmp**(mind->mindname, mind\_name))

        {

            flag = 1;

            break;

        }

    }

**free**(mind);

    mind = NULL;

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("\nFile UserInfo closed error.(from daotujud\_close)\n");

**delay**(3000);

**exit**(1);

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: newmind*

*Function：给对应用户新建一个思维导图的文件*

*Calls:无*

*Called By: new*

*Parameter: account\_name 用户名 新建后将文献信息存储到对应用户名的文件中*

*mind\_name 新建的思维导图文件名*

*dt 暂时存放思维导图信息的结构*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **newmind**(char \*account\_name, char \*mind\_name, MIND \*dt) *//函数类型待定！*

{

    FILE \*fp;

    int i,j; *//框计数变量*

    MINDBOX new\_boxes[HORI][VERT];

    MINDBOX new\_boxestr[HORI][VERTREE];

    FISHINFO new\_fishbone;

    DATEINFO new\_dates;

    char filename[32+ 1];   *//思维导图位置索引*

    char filename\_1[25 + 1]; *//对应用户拥有的导图信息文件索引*

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERT; j ++)

        {

            new\_boxes[i][j].ste = 0;

**strcpy**(new\_boxes[i][j].content, "");

            new\_boxes[i][j].boxcolor = BLACK;

            new\_boxes[i][j].textcolor = BLACK;

        }

    }

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERTREE; j ++)

        {

            new\_boxestr[i][j].ste = 0;

**strcpy**(new\_boxestr[i][j].content, "");

            new\_boxestr[i][j].boxcolor = BLACK;

            new\_boxestr[i][j].textcolor = BLACK;

        }

    }

    for(i = 0; i < BONEMAX; i ++)

    {

**strcpy**(new\_fishbone.bone[i].content,"");

    }

**strcpy**(dt->mindname, mind\_name); *//将名字置入信息暂存区*

**sprintf**(filename, ".\\daotu\\%s\\%s.dat", account\_name, mind\_name);

    if ((fp = **fopen**(filename, "wb+")) == NULL) *//新建一个导图文件*

    {

*//打开失败，异常退出*

**closegraph**();

**printf**("\nFile opening error.(from newmind\_1)\n");

**delay**(3000);

**exit**(1);

    }

    switch(dt->type)

    {

        case 1: *//初始化单向思维导图并保存*

        new\_boxes[0][0].ste = 1;

**strcpy**(new\_boxes[0][0].content,"");

        new\_boxes[0][0].textcolor = BLACK;

        new\_boxes[0][0].boxcolor = BLACK;

        new\_boxes[1][0].ste = 1;

**strcpy**(new\_boxes[1][0].content,"");

        new\_boxes[1][0].textcolor = BLACK;

        new\_boxes[1][0].boxcolor = BLACK;

        new\_boxes[1][1].ste = 1;

**strcpy**(new\_boxes[1][1].content,"");

        new\_boxes[1][1].textcolor = BLACK;

        new\_boxes[1][1].boxcolor = BLACK;

        new\_boxes[1][2].ste = 1;

**strcpy**(new\_boxes[1][2].content,"");

        new\_boxes[1][2].textcolor = BLACK;

        new\_boxes[1][2].boxcolor = BLACK;

**fseek**(fp, 0, 0);

**fwrite**(&(new\_boxes[0][0]), sizeof(MINDBOX), HORI \* VERT, fp);

        break;

        case 2: *//初始化树状图并保存*

        new\_boxestr[0][0].ste = 1;

**strcpy**(new\_boxestr[0][0].content,"");

        new\_boxestr[0][0].textcolor = BLACK;

        new\_boxestr[0][0].boxcolor = BLACK;

        new\_boxestr[0][1].ste = 1;

**strcpy**(new\_boxestr[0][1].content,"");

        new\_boxestr[0][1].textcolor = BLACK;

        new\_boxestr[0][1].boxcolor = BLACK;

        new\_boxestr[1][1].ste = 1;

**strcpy**(new\_boxestr[1][1].content,"");

        new\_boxestr[1][1].textcolor = BLACK;

        new\_boxestr[1][1].boxcolor = BLACK;

        new\_boxestr[2][1].ste = 1;

**strcpy**(new\_boxestr[2][1].content,"");

        new\_boxestr[2][1].textcolor = BLACK;

        new\_boxestr[2][1].boxcolor = BLACK;

**fseek**(fp, 0, 0);

**fwrite**(&(new\_boxestr[0][0]), sizeof(MINDBOX), HORI \* VERT, fp);

        break;

        case 3: *//初始化鱼骨图并保存*

        new\_fishbone.bonenum = 3;

**fseek**(fp, 0, 0);

**fwrite**(&new\_fishbone, sizeof(FISHINFO), 1, fp);

        break;

        case 4: *//初始化时间轴并保存*

        new\_dates.daynum = 0;

**fseek**(fp, 0, 0);

**fwrite**(&new\_dates, sizeof(DATEINFO), 1, fp);

        break;

    }

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("closed error.");

**delay**(3000);

**exit**(1);

    }

**sprintf**(filename\_1, ".\\UserInfo\\%s.dat", account\_name);

    if ((fp = **fopen**(filename\_1, "ab")) == NULL) *//打开用户文件*

    {

*//打开失败，异常退出*

**closegraph**();

**printf**("\nFile opening error.(from newmind\_2)\n");

**delay**(3000);

**exit**(1);

    }

**fwrite**(dt, sizeof(MIND), 1, fp);

    if (**fclose**(fp) != 0) *//关闭文件错误*

    {

**closegraph**();

**printf**("closed error.");

**delay**(3000);

**exit**(1);

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: input\_mind*

*Function：键盘输入显示函数（输入新的思维导图）*

*Calls: 无*

*Called By: new Open.c*

*Parameter: x  输入位置左边界*

*y  输入位置上边界*

*size  字符大小*

*content  输入信息的结构体，详见data.h*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **input\_mind**(int x, int y, int size, ENTER \*content) *//将信息输入到content中，并显示在屏幕上*

{

    char tmp;

    int i;

    if (press == 1)

    {

        if (**mouse\_press**(content->x1, content->y1, content->x2, content->y2) == 1) *//在输入框内点击*

        {

            content->flag = 1;

**clrmous**(MouseX, MouseY);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(content->x1, content->y1, content->x2, content->y2);

**setcolor**(DARKGRAY);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

        }

        else *//在输入框外点击*

        {

            content->flag = 0;

**clrmous**(MouseX, MouseY);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(content->x1, content->y1, content->x2, content->y2);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

        }

    }

    if (content->flag == 1) *//flag为1时表示可以接收键盘输入*

    {

        if (**kbhit**())

        {

            tmp = **getch**();

            if (tmp >= '!' && tmp <= '~') *//输入一个字符*

            {

                if (content->cursor < content->length) *//不能超过最大长度*

                {

                    (content->str)[content->cursor] = tmp;

                    (content->str)[content->cursor + 1] = '\0';

                    (content->cursor)++;

                }

            }

            else if (tmp == '\b') *//删除字符*

            {

                if (content->cursor > 0)

                {

                    (content->str)[content->cursor - 1] = '\0';

                    (content->cursor)--;

                }

            }

**setcolor**(DARKGRAY);

**setlinestyle**(0, 0, 1);

**setfillstyle**(SOLID\_FILL, WHITE);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, size);

**bar**(content->x1 + 2, content->y1 + 2, content->x2 - 2, content->y2 - 2); *//每一步显示之前先用背景色覆盖掉原来信息*

**setcolor**(DARKGRAY);

**outtextxy**(x, y, content->str);

**line**(x + (content->cursor) \* (2 \* size - 2) + 3, y, x + (content->cursor) \* (2 \* size - 2) + 3, y + 2 \* (2 \* size - 2));

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: newdrt*

*Function：新建一个存放新用户导图文件的文件夹*

*Calls: 无*

*Called By: sign\_up*

*Parameter: account\_name 用户名*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **newdrt**(char \*account\_name)

{

    char indexer[15 + 1]; *//新建的文件夹索引*

**sprintf**(indexer, ".\\daotu\\%s", account\_name);

    if (**mkdir**(indexer) != 0) *//新建失败，异常退出*

    {

**closegraph**();

**printf**("fail(from newdrt)");

**delay**(3000);

**exit**(1);

    }

}

9.9 Opena.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  打开界面及功能*

*Function list :  opena   完成打开功能*

*open\_screen  绘制打开界面*

*Attention:*

*Author:  贾田旺*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Opena.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: opena*

*Function：完成打开功能*

*Calls:  input\_mind*

*findmind*

*Called By: operate*

*Parameter: account\_name 账户名*

*daotu\_info  思维导图信息*

*Return: mstate 打开状态*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **opena**(char \*account\_name, MIND \*daotu\_info)

{

    int mstate = 2;               *//表示打开导图状态 2表示仍在页面中 0未打开返回 1表示打开返回*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值*

*// 0 无操作状态 1 取消按钮 2 输入白框 3 打开按钮*

    ENTER mindname = {120 + 128, 100 + 128, 120 + 128 + 144, 100 + 128 + 32, 8, 0, 0, ""};

**clrmous**(MouseX, MouseY);

**delay**(100);

**open\_screen**();

**mouseinit**();

    while (mstate == 2)

    {

**newmouse**(&MouseX, &MouseY, &press);

**input\_mind**(120 + 128 + 4, 100 + 128 + 4, 7, &mindname);

        if (**mouse\_press**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32) == 2) *//取消键且未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32);

            }

        }

        else if (**mouse\_press**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32) == 1) *//取消键且点击*

        {

            mstate = 0;

        }

        else if (**mouse\_press**(120 + 128, 100 + 128, 120 + 128 + 144, 100 + 128 + 32) == 2) *//输入框且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 2; *//光标*

            }

        }

        else if (**mouse\_press**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32) == 2) *//打开键且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(LIGHTRED);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32);

            }

        }

        else if (**mouse\_press**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32) == 1) *//打开键且点击*

        {

*//清除提示信息*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(120 + 128, 100 + 128 + 32 + 2, 120 + 128 + 16 \* 8, 100 + 128 + 32 + 2 + 16);

*//确认是否输入完全*

            if (mindname.str[0] == '\0')

            {

**puthz**(120 + 128, 100 + 128 + 32 + 2, "未输入思维导图名", 16, 16, RED);

            }

*//判断思维导图是否存在*

            else if (!**daotujud**(account\_name, mindname.str))

            {

**puthz**(120 + 128, 100 + 128 + 32 + 2, "该思维导图不存在", 16, 16, RED);

            }

            else

            {

                (\*daotu\_info).type = **findmind**(account\_name, mindname.str).type;

**strcpy**((\*daotu\_info).mindname, **findmind**(account\_name, mindname.str).mindname);

                mstate = 1;

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32);

                break;

            case 2:

                MouseS = 0;

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32);

            }

        }

    }

    return mstate;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: open\_screen*

*Function：绘制打开界面*

*Calls: 无*

*Called By: open*

*Parameter: 无*

*Return: 无*

*Author: 贾田旺*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **open\_screen**(void)

{

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//画框*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(120, 100, 120 + 400, 100 + 280);

**setcolor**(WHITE);

**setlinestyle**(3, 0, 1); *//画破折号线*

**rectangle**(120, 100, 120 + 400, 100 + 280);

**setlinestyle**(1, 0, 3); *//画虚线*

**rectangle**(120 + 2, 100 + 2, 120 + 400 - 2, 100 + 280 - 2);

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(120 + 100, 100 + 28 + 40, 120 + 100 + 24 \* 8 + 8, 100 + 28 + 40 + 24 + 8);

**setlinestyle**(0, 0, 1); *//画破折号线*

**rectangle**(120 + 100, 100 + 28 + 40, 120 + 100 + 24 \* 8 + 8, 100 + 28 + 40 + 24 + 8);

**puthz**(120 + 104, 100 + 32 + 40, "请输入思维导图名", 24, 24, WHITE);

*//输入框*

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(120 + 128, 100 + 128, 120 + 128 + 144, 100 + 128 + 32);

*//打开键*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80, 100 + 192, 120 + 80 + 80, 100 + 192 + 32);

**puthz**(120 + 80 + 8, 100 + 192 + 4, "打开", 24, 40, WHITE);

*//取消键*

**setfillstyle**(SOLID\_FILL, DARKGRAY);

**bar**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(120 + 80 + 80 + 80, 100 + 192, 120 + 80 + 80 + 80 + 80, 100 + 192 + 32);

**puthz**(120 + 80 + 80 + 80 + 8, 100 + 192 + 4, "取消", 24, 40, WHITE);

}

9.10 newa.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:完成新建界面及其功能*

*Function list:   new 新建界面功能*

*new\_screen 绘制新建界面*

*Attention:*

*Author:倪启源*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "newa.h"

#include "OperateT.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: new*

*Function: 新建界面相关信息与跳转功能*

*Calls: input\_mind*

*newmind*

*new\_screen*

*Called by:main*

*Parameter: account\_name 用户名*

*daotu 用于存放指定的思维导图信息*

*Return: mstate 新建状态*

*Author: 倪启源*

*Others：无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **new**(char \*account\_name, MIND \*daotu)

{

    int mstate = 2;               *//表示新建导图状态 2表示仍在页面中 0未新建返回 1表示新建返回*

    MIND new\_mind = {0, ""};      *//新建思维导图信息暂时存放处*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值*

*//0 无操作状态 1 回到空白编辑页按钮*

*//             2 单向导图按钮*

*//             3 树状图按钮*

*//             4 鱼骨图按钮*

*//             5 时间轴按钮*

*// 6 确定框  7 退出框*

    ENTER newmindname = {90, 400, 350, 440, 8, 0, 0, ""};

**clrmous**(MouseX, MouseY);

**delay**(100);

*//mouseinit();*

**new\_screen**();

**mouseinit**();

    while (mstate == 2)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (newmindname.flag != 1)

        {

            while (**bioskey**(1))

            {

**getch**();

            }

        }

**input\_mind**(100, 408, 5, &newmindname);

*//回到编辑页*

        if (**mouse\_press**(540, 448, 610, 468) == 2) *//在按钮上未点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

**puthz**(540, 448, "返回编辑", 16, 18, RED);

**setlinestyle**(SOLID\_FILL, 0, 1);

**setcolor**(RED);

**line**(540, 468, 610, 468);

            }

        }

        else if (**mouse\_press**(540, 448, 610, 468) == 1) *//在返回按钮上且点击*

        {

            mstate = 0;

        }

*//选择某一个导图模板*

        else if (**mouse\_press**(50, 70, 200, 200) == 2) *//在单向导图未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if (**mouse\_press**(50, 70, 200, 200) == 1) *//在单向导图且点击*

        {

**puthz**(60, 176, "单向思维导图", 16, 18, WHITE); *//单向导图*

**puthz**(230, 176, "树状图", 16, 18, BLACK);     *//树状图*

**puthz**(400, 176, "鱼骨图", 16, 18, BLACK);     *//鱼骨图*

**puthz**(60, 326, "时间轴", 16, 18, BLACK);      *//时间轴*

            new\_mind.type = 1;

        }

        else if (**mouse\_press**(220, 70, 370, 200) == 2) *//在树状图未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if (**mouse\_press**(220, 70, 370, 200) == 1) *//在树状图且点击*

        {

**puthz**(60, 176, "单向思维导图", 16, 18, BLACK); *//单向导图*

**puthz**(230, 176, "树状图", 16, 18, WHITE);     *//树状图*

**puthz**(400, 176, "鱼骨图", 16, 18, BLACK);     *//鱼骨图*

**puthz**(60, 326, "时间轴", 16, 18, BLACK);      *//时间轴*

            new\_mind.type = 2;

        }

        else if (**mouse\_press**(390, 70, 540, 200) == 2) *//在鱼骨图未点击*

        {

            pre\_state = state;

            state = 4;

            if (pre\_state != 4) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if (**mouse\_press**(390, 70, 540, 200) == 1) *//在鱼骨图且点击*

        {

**puthz**(60, 176, "单向思维导图", 16, 18, BLACK); *//单向导图*

**puthz**(230, 176, "树状图", 16, 18, BLACK);     *//树状图*

**puthz**(400, 176, "鱼骨图", 16, 18, WHITE);     *//鱼骨图*

**puthz**(60, 326, "时间轴", 16, 18, BLACK);      *//时间轴*

            new\_mind.type = 3;

        }

        else if (**mouse\_press**(50, 220, 200, 350) == 2) *//在时间轴未点击*

        {

            pre\_state = state;

            state = 5;

            if (pre\_state != 5) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if (**mouse\_press**(50, 220, 200, 350) == 1) *//在时间轴且点击*

        {

**puthz**(60, 176, "单向思维导图", 16, 18, BLACK); *//单向导图*

**puthz**(230, 176, "树状图", 16, 18, BLACK);     *//树状图*

**puthz**(400, 176, "鱼骨图", 16, 18, BLACK);     *//鱼骨图*

**puthz**(60, 326, "时间轴", 16, 18, WHITE);      *//时间轴*

            new\_mind.type = 4;

        }

        else if (**mouse\_press**(360, 400, 420, 440) == 2) *//在确定框未点击*

        {

            pre\_state = state;

            state = 6;

            if (pre\_state != 6) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if (**mouse\_press**(360, 400, 420, 440) == 1) *//在确定框且点击*

        {

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(95, 444, 350, 470);

*//确认是否符合要求*

            if (newmindname.str[0] == '\0') *//有无？*

            {

**puthz**(100, 448, "未输入新建文件名", 16, 18, LIGHTGRAY);

            }

            else if (**strlen**(newmindname.str) >= 9) *//溢出？*

            {

**puthz**(100, 448, "最多八字符", 16, 18, LIGHTGRAY);

            }

            else if (new\_mind.type == 0) *//种类选了？*

            {

**puthz**(100, 448, "选个种类吧", 16, 18, LIGHTGRAY);

            }

            else if (**daotujud**(account\_name, newmindname.str)) *//重复？*

            {

**puthz**(100, 448, "文件名已存在", 16, 18, LIGHTGRAY);

            }

            else

            {

**newmind**(account\_name, newmindname.str, &new\_mind);

                \*daotu = new\_mind;

                mstate = 1;

            }

        }

        else if (**mouse\_press**(430, 448, 464, 468) == 2)

        {

            pre\_state = state;

            state = 7;

            if (pre\_state != 7) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

**puthz**(430, 448, "退出", 16, 18, RED);

**setlinestyle**(SOLID\_FILL, 0, 1);

**setcolor**(RED);

**line**(430, 448, 464, 468);

            }

        }

        else if (**mouse\_press**(430, 448, 464, 468) == 1) *//退出框*

        {

**closegraph**(); *//关闭画图*

**delay**(100);   *//可解决重启动时鼠标状态不稳定，但有学长反应容易崩溃，慎用*

**exit**(0);      *//退出程序*

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

*//如果状态发生改变，且之前不是无操作状态，则可能需清除标亮*

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**puthz**(540, 448, "返回编辑", 16, 18, BLUE);

**setlinestyle**(SOLID\_FILL, 0, 1);

**setcolor**(BLUE);

**line**(540, 468, 610, 468);

                break;

            case 2:

            case 3:

            case 4:

            case 5:

            case 6:

                MouseS = 0;

                break;

            case 7:

                MouseS = 0;

**puthz**(430, 448, "退出", 16, 18, BLUE);

**setlinestyle**(SOLID\_FILL, 0, 1);

**setcolor**(BLUE);

**line**(430, 448, 464, 468);

                break;

            }

        }

    }

    return mstate;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: new\_screen*

*Function：画新建界面*

*Calls: 无*

*Called By: new*

*Parameter：无*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **new\_screen**(void)

{

    int i, j; *//画框计数用*

    int head[8] = {465, 75, 440, 95, 490, 95, 465, 75};

    int tail[8] = {465, 145, 440, 165, 490, 165, 465, 145};

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(0, 0, 640, 480);

*//返回编辑，回到空白编辑页按钮*

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**puthz**(540, 448, "返回编辑", 16, 18, BLUE);

**setlinestyle**(SOLID\_LINE, 0, 1);

**setcolor**(BLUE);

**line**(540, 468, 610, 468);

*//选择一个开始吧*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**setcolor**(LIGHTGRAY);

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**bar**(0, 0, 640, 60);

**puthz**(30, 30, "选择一个开始吧", 24, 30, BLACK);

*//思维导图模板种类*

*//置图框*

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(50, 70, 200, 170);  *//单向导图*

**bar**(220, 70, 370, 170); *//树状图*

**bar**(390, 70, 540, 170); *//鱼骨图*

**bar**(50, 220, 200, 320); *//时间轴*

*//名称框*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(50, 170, 200, 200);  *//单向导图*

**bar**(220, 170, 370, 200); *//树状图*

**bar**(390, 170, 540, 200); *//鱼骨图*

**bar**(50, 320, 200, 350);  *//时间轴*

*//名称*

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**puthz**(60, 176, "单向思维导图", 16, 18, BLACK); *//单向导图*

**puthz**(230, 176, "树状图", 16, 18, BLACK);     *//树状图*

**puthz**(400, 176, "鱼骨图", 16, 18, BLACK);     *//鱼骨图*

**puthz**(60, 326, "时间轴", 16, 18, BLACK);      *//时间轴*

*//图样*

*//框部分*

**setfillstyle**(SOLID\_FILL, LIGHTBLUE);

**setcolor**(LIGHTGREEN);

**setlinestyle**(0, 0, 1);

**bar**(60, 110, 110, 130); *//单向导图*

    for (i = 0; i < 3; i++)

    {

**bar**(140, 80 + i \* 30, 190, 100 + i \* 30);

    }

**bar**(270, 80, 320, 100); *//树状图*

    for (i = 0; i < 3; i++)

    {

**bar**(235 + i \* 45, 120, 265 + i \* 45, 140);

    }

    for (i = 0; i < 3; i++) *//时间轴*

    {

**bar**(50 + 10 + i \* 50, 250, 50 + 10 + 30 + i \* 50, 290);

        for (j = 0; j < 3; j++)

        {

**setcolor**(BLACK);

**line**(50 + 10 + i \* 50, 260 + j \* 10, 50 + 10 + 30 + i \* 50, 260 + j \* 10);

        }

    }

*//鱼骨图*

**fillpoly**(4, head);

**fillpoly**(4, tail);

*//连线部分*

**setcolor**(BLACK);

**setlinestyle**(SOLID\_LINE, 0, 2);

    for (i = 0; i < 3; i++) *//单向导图*

    {

**line**(110, 120, 140, 90 + i \* 30);

    }

    for (i = 0; i < 3; i++) *//树状图*

    {

**line**(295, 100, 250 + i \* 45, 120);

    }

    for (i = 0; i < 2; i++) *//时间轴*

    {

**line**(50 + 10 + 30 + i \* 50, 270, 50 + 10 + 30 + 20 + i \* 50, 270);

    }

*//鱼骨图*

**line**(465, 95, 465, 145);

**line**(465, 105, 450, 120);

**line**(450, 120, 425, 120);

**line**(465, 120, 480, 135);

**line**(480, 135, 505, 135);

*//退出按钮*

**settextjustify**(LEFT\_TEXT, TOP\_TEXT);

**puthz**(430, 448, "退出", 16, 18, BLUE);

**setlinestyle**(SOLID\_LINE, 0, 1);

**setcolor**(BLUE);

**line**(430, 468, 464, 468);

*//输入导图文件名字框*

**setlinestyle**(SOLID\_LINE, 0, 5);

**setcolor**(DARKGRAY);

**setfillstyle**(SOLID\_FILL, BLUE);

**bar**(10, 400, 90, 440);

**rectangle**(10, 400, 90, 440);

**puthz**(20, 408, "导图名", 16, 20, YELLOW);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(90, 400, 350, 440); *//输入框*

**setfillstyle**(SOLID\_FILL, RED);

**bar**(360, 400, 420, 440);

**puthz**(370, 408, "确定", 16, 30, YELLOW);

}

9.11 Edit.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  具体编辑思维导图功能及界面*

*Function list :  edit 编辑页面功能*

*edit\_screen 绘制编辑页面*

*color\_screen 绘制调色板界面*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Edit.h"

#include <graphics.h>

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edit*

*Function：编辑界面功能*

*Calls: edit\_screen*

*findbox*

*drawmind*

*addbox*

*deletebox*

*edit\_mind*

*Called By: operate*

*Parameter: editp*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edit**(**MINDBOX** (\*editp)[VERT])

{

    int e\_state = 1;              *//表示是否在edit页面内 1为仍在 0为跳转    控制页面跳转*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态*

*// 1 完成键 2 新增键 3 删除键 4 保存键*

    int heng, zong;

    MINPUT minput = {7, 0, ""};

    int find\_flag = 0; *//表示是否找到框*

    int colortemp = WHITE; *//暂存当前选中的颜色 没有选择时，置为WHITE*

**clrmous**(MouseX, MouseY);

**delay**(100);

**edit\_screen**();

**color\_screen**();

**drawmind**(editp);

**mouseinit**();

    while (e\_state == 1)

    {

**newmouse**(&MouseX, &MouseY, &press);

        while (**bioskey**(1))

        {

**getch**();

        }

        if (**mouse\_press**(640 - 42, 0, 640, 24) == 2) *//完成且未点击 未制作点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

            }

        }

        else if (**mouse\_press**(640 - 42, 0, 640, 24) == 1) *//完成且点击*

        {

            e\_state = 0;

**delay**(100);

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 2) *//新增且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 1) *//新增且点击*

        {

            if (heng < HORI && find\_flag == 1)

            {

**addbox**(editp, heng, zong);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 2) *//删除且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 1) *//删除且点击*

        {

            if (find\_flag && (heng + zong) != 0)

            {

**deletebox**(editp, heng, zong);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(0, 24, 496, 480) == 1) *//选要编辑的框*

        {

            find\_flag = **findbox**(editp, &heng, &zong);

            if (find\_flag == 1)

            {

**clrmous**(MouseX, MouseY);

**delay**(25); *//画图前先清除鼠标，防止干扰*

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(RED);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(8 + heng \* 80, 28 + zong \* 24, 8 + heng \* 80 + 64, 28 + zong \* 24 + 16);

**rectangle**(8 + heng \* 80, 28 + zong \* 24, 8 + heng \* 80 + 64, 28 + zong \* 24 + 16);

**edit\_mind**(8 + heng \* 80, 28 + zong \* 24, &minput, &editp[heng][zong],BLACK);

**strcpy**(minput.str, "");

                minput.cursor = 0;

            }

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

**setcolor**(RED);

**setlinestyle**(0, 0, 3);

            if(find\_flag == 1)

            {

**rectangle**(8 + heng \* 80, 28 + zong \* 24, 8 + heng \* 80 + 64, 28 + zong \* 24 + 16);

            }

        }

        else if(**mouse\_press**(504 + 17, 280, 604 + 17, 300) == 2) *//在颜色框未点击*

        {

            pre\_state = state;

            state = 5;

            if (pre\_state != 5) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if(**mouse\_press**(504 + 17, 280, 604 + 17, 300) == 1) *//在颜色框且点击*

        {

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

            if(**mouse\_press**(504 + 17, 280, 524 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = BLACK;

**rectangle**(504 + 17, 280, 524 + 17, 300);

            }

            else if(**mouse\_press**(524 + 17, 280, 544 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = CYAN;

**rectangle**(524 + 17, 280, 544 + 17, 300);

            }

            else if(**mouse\_press**(544 + 17, 280, 564 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = GREEN;

**rectangle**(544 + 17, 280, 564 + 17, 300);

            }

            else if(**mouse\_press**(564 + 17, 280, 584 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = BLUE;

**rectangle**(564 + 17, 280, 584 + 17, 300);

            }

            else if(**mouse\_press**(584 + 17, 280, 604 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = LIGHTGRAY;

**rectangle**(584 + 17, 280, 604 + 17, 300);

            }

        }

        else if(**mouse\_press**(496 + 25, 248, 496 + 25 + 42, 272) == 2) *//字色未点击*

        {

            pre\_state = state;

            state = 6;

            if (pre\_state != 6) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 248, 496 + 25 + 42, 272);

            }

        }

        else if(**mouse\_press**(496 + 25, 248, 496 + 25 + 42, 272) == 1) *//字色且点击*

        {

            if(find\_flag == 1 && colortemp != WHITE)

            {

                editp[heng][zong].textcolor = colortemp;

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**color\_screen**();

                colortemp = WHITE;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

                find\_flag = 0;

            }

        }

        else if(**mouse\_press**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272) == 2) *//框色未点击*

        {

            pre\_state = state;

            state = 7;

            if (pre\_state != 7) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

            }

        }

        else if(**mouse\_press**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272) == 1) *//框色且点击*

        {

            if(find\_flag == 1 && colortemp != WHITE)

            {

                editp[heng][zong].boxcolor = colortemp;

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**color\_screen**();

                colortemp = WHITE;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

                find\_flag = 0;

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

                break;

            case 4:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8 + 24 + 8, 496 + 25 + 42, 24 + 8 + 24 + 8 + 24);

                break;

            case 5:

                MouseS = 0;

                break;

            case 6:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 248, 496 + 25 + 42, 272);

                break;

            case 7:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

                break;

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edit\_screen*

*Function：绘制编辑界面*

*Calls: 无*

*Called By: edit*

*edittr*

*editfb*

*Parameter: 无*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edit\_screen**(void)

{

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 0, 496, 480);

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(496, 0, 640, 480);

**bar**(0, 0, 496, 24);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**line**(496, 0, 496, 480);

**line**(496, 240, 640, 240);

*//画框*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

**puthz**(496 + 25 + 4, 24 + 8 + 4, "新增", 16, 18, WHITE);

**bar**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

**puthz**(496 + 25 + 42 + 10 + 4, 24 + 8 + 4, "删除", 16, 18, WHITE);

**bar**(496 + 25, 248, 496 + 25 + 42, 272);

**puthz**(496 + 25 + 4, 248 + 4, "字色", 16, 18, WHITE);

**bar**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

**puthz**(496 + 25 + 42 + 10 + 4, 248 + 4, "框色", 16, 18, WHITE);

**bar**(640 - 42, 0, 640, 24);

**puthz**(640 - 42 + 4, 0 + 4, "完成", 16, 18, LIGHTRED);

**setlinestyle**(0, 0, 1);

**setcolor**(WHITE);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

**rectangle**(496 + 25, 248, 496 + 25 + 42, 272);

**rectangle**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

**rectangle**(640 - 42, 0, 640, 24);

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: color\_screen*

*Function：绘制调色板界面*

*Calls: 无*

*Called By: edit*

*edittr*

*Parameter: 无*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **color\_screen**(void)

{

*//画调色板*

**setcolor**(BLACK);

**setlinestyle**(0, 0, 1);

**rectangle**(504 + 17 - 1, 280 - 1, 604 + 17 + 1, 300 + 1);

**setfillstyle**(SOLID\_FILL, BLACK);

**bar**(504 + 17, 280, 524 + 17, 300);

**setfillstyle**(SOLID\_FILL, CYAN);

**bar**(524 + 17, 280, 544 + 17, 300);

**setfillstyle**(SOLID\_FILL, GREEN);

**bar**(544 + 17, 280, 564 + 17, 300);

**setfillstyle**(SOLID\_FILL, BLUE);

**bar**(564 + 17, 280, 584 + 17, 300);

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(584 + 17, 280, 604 + 17, 300);

}

9.12 Boxa.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑页面框图操作（查、画、增）函数集*

*Function list :  findbox 寻找单向思维导图的框*

*drawmind 画出当前的单向思维导图*

*addbox 新增，新增单向思维导图的一个子节点*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxa.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: findbox*

*Function：寻找单向思维导图的框*

*Calls: drawmind*

*Called By: edit*

*Parameter: editp 受选择的数组*

*ho 水平位置*

*ve 竖直位置*

*Return：flag 是否找到框*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **findbox**(**MINDBOX** (\*editp)[VERT], int \*ho, int \*ve)

{

    int i,j; *//计数变量*

    int flag = 0;

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERT; j ++)

        {

            if(**mouse\_press**(8 + i \* 80, 28 + j \* 24, 8 + i \* 80 + 64, 28 + j \* 24 + 16) == 1)

            {

                if(editp[i][j].ste == 1)

                {

                    \*ho = i;

                    \*ve = j;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind**(editp);

**setcolor**(RED);

**setlinestyle**(SOLID\_LINE, 0, 3);

**rectangle**(8 + i \* 80, 28 + j \* 24, 8 + i \* 80 + 64, 28 + j \* 24 + 16);

                    flag = 1;

                }

                break;

            }

        }

        if(flag == 1)

        {

            break;

        }

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: drawmind*

*Function：画出当前的单向思维导图*

*Calls: 无*

*Called By: edit*

*Parameter: editp 当前的存储思维导图的数组*

*Return：无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **drawmind**(**MINDBOX** (\*editp)[VERT])

{

    int i, j; *//计数变量*

    int k; *//上溯计数变量*

**setlinestyle**(SOLID\_LINE, 0, 3);

**setfillstyle**(SOLID\_FILL, WHITE);

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERT; j ++)

        {

            if(editp[i][j].ste == 1)

            {

**setcolor**(editp[i][j].boxcolor);

**rectangle**(8 + i \* 80, 28 + j \* 24, 8 + i \* 80 + 64, 28 + j \* 24 + 16);

**bar**(8 + i \* 80, 28 + j \* 24, 8 + i \* 80 + 64, 28 + j \* 24 + 16);

*//settextjustify(LEFT\_TEXT, TOP\_TEXT);*

**setcolor**(editp[i][j].textcolor);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(8 + i \* 80 + 6 , 28 + j \* 24, editp[i][j].content);

                if(i != 0 || j != 0)

                {

                    k = j;

                    while(editp[i - 1][k].ste == 0)

                    {

                        k --;

                    }

**setcolor**(BLACK);

**line**(8 + (i - 1) \* 80 + 64, 28 + k \* 24 + 8, 8 + i \* 80, 28 + j \* 24 + 8);

                }

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: addbox*

*Function：新增，新增单向思维导图的一个子节点*

*Calls: movebox*

*Called By: edit*

*Parameter: editp 当前的存储思维导图的数组*

*ho 水平位置*

*ve 竖直位置*

*Return: 无*

*Author: 倪启源*

*Others: 判断新增的情况主要可以分为三种情况：*

*1，选中的框的ho值为5，这个时候会溢出，点击addbox无反应*

*（判断在函数调用之前完成）*

*2，选中的框ho值小于5*

*（1）如果该框（ho，ve）之后的（ho+1，ve）的ste值等于0，*

*则直接新增在这个框即可*

*（2）从纵坐标ve+1开始遍历，每一行从最右边开始遍历，如果*

*遍历到ho+1仍然ste=0，则直接从这一行开始的所有行下移一行*

*在这一行的横坐标ho+1处新增一个框。*

*若遇到ste=1的情况，则判断该行的横坐标ho处有无ste=1*

*a.ste=1，则这一行的横坐标ho+1处为新增的框*

*b.ste=0，开始遍历下一行*

*注意（2）情况一定要先检查看看纵坐标VERT-1处有无ste=1的框，*

*若有，则不执行任何操作（无法新增行！）*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **addbox**(**MINDBOX** (\*editp)[VERT], int ho, int ve) *//还有微量bug，待de*

{

    int i, j, k; *//计数变量*

    int movei, movej; *//移动计数变量*

    int check; *//检测最后一排是否已经有元素 若check=HORI，*

*//则最后一行没有框*

*//int states; //框的修改方式存储变量*

    int flag = 0;

    if(ho == HORI - 1)

    {

        return;

    }

    for(check = 0; check < HORI; check ++)

    {

        if(editp[check][VERT - 1].ste == 1)

        {

            break;

        }

    }

    if(editp[ho + 1][ve].ste == 0)

    {

        editp[ho + 1][ve].ste = 1;

    }

    else

    {

        for(j = ve + 1; j < VERT; j ++)

        {

            for(i = HORI - 1; i > ho; i --)

            {

                if(editp[i][j].ste == 1 && i > ho)

                {

                    break;

                }

            }

            if(i == ho || editp[ho][j].ste == 1) *//新增的框在（ho+1，j）处*

            {

                if(check == HORI)

                {

                    for(movej = VERT - 2; movej >= j; movej --)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**movebox**(&editp[movei][movej + 1], &editp[movei][movej]);

                        }

                    }

                    editp[ho + 1][j].ste = 1;

                }

                flag = 1;

            }

            if(flag == 1)

            {

                break;

            }

        }

    }

}

9.13 Boxadl.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  思维导图删框函数 基础函数（移动框，清空框）*

*Function list :  deletebox 删除，将所有的子节点一并删除，并对其余节点的坐标进行调整*

*movebox 移动信息函数*

*vanbox 置空信息函数*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxadl.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: deletebox*

*Function：删除，将所有的子节点一并删除，并对其余节点的坐标进行调整*

*Calls: movebox*

*vanbox*

*Called By: edit*

*Parameter: editp 当前的存储思维导图的数组*

*ho 水平位置*

*ve 竖直位置*

*Return: 无*

*Author: 倪启源*

*Others: 判断删除的情况：*

*先遍历本行，如果出现的第一个框是自己，记state=1*

*否则记state=0*

*从下一行开始遍历，如果第一框在横坐标在本框之前*

*state=1全体上移至ve*

*state=0全体上移至ve+1*

*如果第一框坐标等于本框*

*state=1全体上移至ve*

*state=0全体上移至ve，本行只从横坐标ho开始移动*

*如果第一框出现在本框之后*

*继续下一轮循环*

*如果无框*

*state=1清空ve行开始的所有*

*state=0清空ve+1行开始的所有以及ve行ho及以后的所有*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **deletebox**(**MINDBOX** (\*editp)[VERT], int ho, int ve)

{

    int benhang; *//遍历本行所使用的变量*

    int i, j; *//横纵遍历*

    int movei, movej; *//移动框工具变量*

    for(benhang = 0; benhang < ho; benhang ++)

    {

        if(editp[benhang][ve].ste == 1)

        {

            break;

        }

    }

    if(ve == VERT - 1)

    {

        for(movei = ho; movei < HORI; movei ++)

        {

**vanbox**(&editp[movei][ve]);

        }

        return;

    }

    for(j = ve + 1; j < VERT; j ++)

    {

        for(i = 0; i < HORI; i ++)

        {

            if(editp[i][j].ste == 1)

            {

                break;

            }

        }

        if(i < ho)

        {

            if(benhang == ho)

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][ve]);

                }

                if(j > ve + 1)

                {

                    for(movej = ve + 1; movej < j; movej ++)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movej = j; movej < VERT; movej ++)

                {

                    for(movei = 0; movei < HORI; movei ++)

                    {

**movebox**(&editp[movei][ve + movej - j], &editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][ve]);

                }

                if(j > ve + 1)

                {

                    for(movej = ve + 1; movej < j; movej ++)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                    for(movej = j; movej < VERT; movej ++)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**movebox**(&editp[movei][ve + movej - j + 1], &editp[movei][movej]);

                        }

                    }

                }

            }

            break;

        }

        else if(i == ho)

        {

            if(benhang == ho)

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][ve]);

                }

                if(j > ve + 1)

                {

                    for(movej = ve + 1; movej < j; movej ++)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movej = j; movej < VERT; movej ++)

                {

                    for(movei = 0; movei < HORI; movei ++)

                    {

**movebox**(&editp[movei][ve + movej - j], &editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**movebox**(&editp[movei][ve], &editp[movei][j]);

                }

                if(j > ve + 1)

                {

                    for(movej = ve + 1; movej < j; movej ++)

                    {

                        for(movei = 0; movei < HORI; movei ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movej = j + 1; movej < VERT; movej ++)

                {

                    for(movei = 0; movei < HORI; movei ++)

                    {

**movebox**(&editp[movei][ve + movej - j], &editp[movei][movej]);

                    }

                }

            }

            break;

        }

        else if(i < HORI && i > ho)

        {

            continue;

        }

        else

        {

            if(benhang == ho)

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][ve]);

                }

                for(movej = ve + 1; movej < VERT; movej ++)

                {

                    for(movei = 0; movei < HORI; movei ++)

                    {

**vanbox**(&editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movei = ho; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][ve]);

                }

                for(movej = ve + 1; movej < VERT; movej ++)

                {

                    for(movei = 0; movei < HORI; movei ++)

                    {

**vanbox**(&editp[movei][movej]);

                    }

                }

            }

            break;

        }

    }

    if(j == VERT)

    {

        for(movei = ho; movei < HORI; movei ++)

        {

**vanbox**(&editp[movei][ve]);

        }

        if(ve + 1 < VERT)

        {

            for(movej = ve + 1; movej < VERT; movej ++)

            {

                for(movei = 0; movei < HORI; movei ++)

                {

**vanbox**(&editp[movei][movej]);

                }

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: movebox*

*Function：移动信息函数*

*Calls:*

*Called By: Boxa.c Boxadl.c Boxtr.c Boxtrdl.c*

*Parameter: former 接收数据的单元*

*latter 释放数据的单元*

*Return: 无*

*Author: 倪启源*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **movebox**(MINDBOX \*former, MINDBOX \*latter)

{

    former -> ste = latter -> ste;

    former -> boxcolor = latter -> boxcolor;

    former -> textcolor =latter -> textcolor;

    latter -> ste = 0;

    latter -> boxcolor = BLACK;

    latter -> textcolor = BLACK;

**strcpy**(former -> content, latter -> content);

**strcpy**(latter -> content,"");

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: vanbox*

*Function：置空信息函数*

*Calls:*

*Called By: Boxa.c Boxadl.c Boxtr.c Boxtrdl.c*

*Parameter: clrbox 清除数据的单元*

*Return: 无*

*Author: 倪启源*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **vanbox**(MINDBOX \*clrbox)

{

    clrbox -> ste = 0;

**strcpy**(clrbox -> content, "");

    clrbox -> boxcolor = BLACK;

    clrbox -> textcolor = BLACK;

}

9.14 Edittr.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  具体编辑树状图功能及界面*

*Function list :  edittr 编辑页面功能*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Edit.h"

#include "Edittr.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edittr*

*Function：编辑树状图界面功能*

*Calls: edit\_screen*

*Called By: operate*

*Parameter: 无*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edittr**(**MINDBOX** (\*editp)[VERTREE])

{

    int e\_state = 1;              *//表示是否在edit页面内 1为仍在 0为跳转    控制页面跳转*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态*

*// 1 完成键 2 新增键 3 删除键 4 保存键*

    int heng, zong;

    MINPUT minput = {7, 0, ""};

    int find\_flag = 0; *//表示是否找到框*

    int colortemp = WHITE; *//暂存当前选中的颜色 没有选择时，置为WHITE*

**clrmous**(MouseX, MouseY);

**delay**(100);

**edit\_screen**();

**color\_screen**();

**drawmind\_tr**(editp);

**mouseinit**();

    while (e\_state == 1)

    {

**newmouse**(&MouseX, &MouseY, &press);

        while (**bioskey**(1))

        {

**getch**();

        }

        if (**mouse\_press**(640 - 42, 0, 640, 24) == 2) *//完成且未点击 未制作点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

            }

        }

        else if (**mouse\_press**(640 - 42, 0, 640, 24) == 1) *//完成且点击*

        {

            e\_state = 0;

**delay**(100);

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 2) *//新增且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 1) *//新增且点击*

        {

            if (heng < HORI && find\_flag == 1)

            {

**addbox\_tr**(editp, heng, zong);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 2) *//删除且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 1) *//删除且点击*

        {

            if (find\_flag && (heng + zong) != 0)

            {

**deletebox\_tr**(editp, heng, zong);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(0, 24, 496, 480) == 1) *//选要编辑的框*

        {

            find\_flag = **findbox\_tr**(editp, &heng, &zong);

            if (find\_flag == 1)

            {

**clrmous**(MouseX, MouseY);

**delay**(25); *//画图前先清除鼠标，防止干扰*

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(RED);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(8 + heng \* 80, 34 + zong \* 36, 8 + heng \* 80 + 64, 34 + zong \* 36 + 16);

**rectangle**(8 + heng \* 80, 34 + zong \* 36, 8 + heng \* 80 + 64, 34 + zong \* 36 + 16);

**edit\_mind**(8 + heng \* 80, 34 + zong \* 36, &minput, &editp[heng][zong],BLACK);

**strcpy**(minput.str, "");

                minput.cursor = 0;

            }

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

**setcolor**(RED);

**setlinestyle**(0, 0, 3);

            if(find\_flag == 1)

            {

**rectangle**(8 + heng \* 80, 34 + zong \* 36, 8 + heng \* 80 + 64, 34 + zong \* 36 + 16);

            }

        }

        else if(**mouse\_press**(504 + 17, 280, 604 + 17, 300) == 2) *//在颜色框未点击*

        {

            pre\_state = state;

            state = 5;

            if (pre\_state != 5) *//防止被重复标亮（只有之前是其它状态才标亮）*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1;

            }

        }

        else if(**mouse\_press**(504 + 17, 280, 604 + 17, 300) == 1) *//在颜色框且点击*

        {

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

            if(**mouse\_press**(504 + 17, 280, 524 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = BLACK;

**rectangle**(504 + 17, 280, 524 + 17, 300);

            }

            else if(**mouse\_press**(524 + 17, 280, 544 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = CYAN;

**rectangle**(524 + 17, 280, 544 + 17, 300);

            }

            else if(**mouse\_press**(544 + 17, 280, 564 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = GREEN;

**rectangle**(544 + 17, 280, 564 + 17, 300);

            }

            else if(**mouse\_press**(564 + 17, 280, 584 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = BLUE;

**rectangle**(564 + 17, 280, 584 + 17, 300);

            }

            else if(**mouse\_press**(584 + 17, 280, 604 + 17, 300) == 1)

            {

**color\_screen**();

                colortemp = LIGHTGRAY;

**rectangle**(584 + 17, 280, 604 + 17, 300);

            }

        }

        else if(**mouse\_press**(496 + 25, 248, 496 + 25 + 42, 272) == 2) *//字色未点击*

        {

            pre\_state = state;

            state = 6;

            if (pre\_state != 6) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 248, 496 + 25 + 42, 272);

            }

        }

        else if(**mouse\_press**(496 + 25, 248, 496 + 25 + 42, 272) == 1) *//字色且点击*

        {

            if(find\_flag == 1 && colortemp != WHITE)

            {

                editp[heng][zong].textcolor = colortemp;

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**color\_screen**();

                colortemp = WHITE;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

                find\_flag = 0;

            }

        }

        else if(**mouse\_press**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272) == 2) *//框色未点击*

        {

            pre\_state = state;

            state = 7;

            if (pre\_state != 7) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

            }

        }

        else if(**mouse\_press**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272) == 1) *//框色且点击*

        {

            if(find\_flag == 1 && colortemp != WHITE)

            {

                editp[heng][zong].boxcolor = colortemp;

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(504 + 17 - 2, 280 - 2, 604 + 17 + 2, 300 + 2);

**color\_screen**();

                colortemp = WHITE;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

                find\_flag = 0;

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

                break;

            case 4:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8 + 24 + 8, 496 + 25 + 42, 24 + 8 + 24 + 8 + 24);

                break;

            case 5:

                MouseS = 0;

                break;

            case 6:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 248, 496 + 25 + 42, 272);

                break;

            case 7:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 248, 496 + 25 + 42 + 10 + 42, 272);

                break;

            }

        }

    }

}

9.15 Boxtr.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑树状图操作（查、画、增）函数集*

*Function list :  findbox\_tr 寻找树状图的框*

*drawmind\_tr 画出当前的树状图*

*addbox\_tr 新增，新增树状图一个子节点*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxtr.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: findbox\_tr*

*Function：寻找树状图的框*

*Calls: drawmind\_tr*

*Called By: edittr*

*Parameter: editp 受选择的数组*

*ho 水平位置*

*ve 竖直位置*

*Return：flag 是否找到框*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **findbox\_tr**(**MINDBOX** (\*editp)[VERTREE], int \*ho, int \*ve)

{

    int i,j; *//计数变量*

    int flag = 0; *//flag代表是否选中了所点击的区域，若选中了，则变为1，否则为0*

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERTREE; j ++)

        {

            if(**mouse\_press**(8 + i \* 80, 34 + j \* 36, 8 + i \* 80 + 64, 34 + j \* 36 + 16) == 1)

            {

                if(editp[i][j].ste == 1)

                {

                    \*ho = i;

                    \*ve = j;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawmind\_tr**(editp);

**setcolor**(RED);

**setlinestyle**(SOLID\_LINE, 0, 3);

**rectangle**(8 + i \* 80, 34 + j \* 36, 8 + i \* 80 + 64, 34 + j \* 36 + 16);

                    flag = 1;

                }

                break;

            }

        }

        if(flag == 1)

        {

            break;

        }

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: drawmind\_tr*

*Function：画出当前的树状图*

*Calls: 无*

*Called By: edittr*

*Parameter: editp 当前的存储树状图的数组*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **drawmind\_tr**(**MINDBOX** (\*editp)[VERTREE])

{

    int i, j; *//计数变量*

    int k; *//上溯计数变量*

**setlinestyle**(SOLID\_LINE, 0, 3);

**setfillstyle**(SOLID\_FILL, WHITE);

    for(i = 0; i < HORI; i ++)

    {

        for(j = 0; j < VERTREE; j ++)

        {

            if(editp[i][j].ste == 1)

            {

**setcolor**(editp[i][j].boxcolor);

**rectangle**(8 + i \* 80, 34 + j \* 36, 8 + i \* 80 + 64, 34 + j \* 36 + 16);

**bar**(8 + i \* 80, 34 + j \* 36, 8 + i \* 80 + 64, 34 + j \* 36 + 16);

*//settextjustify(LEFT\_TEXT, TOP\_TEXT);*

**setcolor**(editp[i][j].textcolor);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(8 + i \* 80 + 6 , 34 + j \* 36, editp[i][j].content);

                if(i != 0 || j != 0)

                {

                    k = i;

                    while(editp[k][j - 1].ste == 0)

                    {

                        k --;

                    }

**setcolor**(BLACK);

**line**(8 + k \* 80 + 32, 34 + (j - 1) \* 36 + 16, 8 + i \* 80 + 32, 34 + j \* 36);

                }

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: addbox\_tr*

*Function：新增，新增树状图一个子节点*

*Calls: movebox*

*Called By: edit.c*

*Parameter: editp 要画出的导图*

*ho 水平位置*

*ve 竖直位置*

*Return: 无*

*Author: 倪启源*

*Others: 具体思路可参考单向思维导图*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **addbox\_tr**(**MINDBOX** (\*editp)[VERTREE], int ho, int ve) *//还有微量bug，待de*

{

    int i, j, k; *//计数变量*

    int movei, movej; *//移动计数变量*

    int check; *//检测最后一排是否已经有元素 若check=HORI，*

*//则最后一行没有框*

    int states; *//框的修改方式存储变量*

    int flag = 0;

    if(ve == VERTREE - 1)

    {

        return;

    }

    for(check = 0; check < VERTREE; check ++)

    {

        if(editp[HORI - 1][ve].ste == 1)

        {

            break;

        }

    }

    if(editp[ho][ve + 1].ste == 0)

    {

        editp[ho][ve + 1].ste = 1;

    }

    else

    {

        for(i = ho + 1; i < HORI; i ++)

        {

            for(j = VERTREE - 1; j > ve; j --)

            {

                if(editp[i][j].ste == 1 && j > ve)

                {

                    break;

                }

            }

            if(j == ve || editp[i][ve].ste == 1) *//新增的框在（ho+1，j）处*

            {

                if(check == VERTREE)

                {

                    for(movei = HORI - 2; movei >= i; movei --)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**movebox**(&editp[movei + 1][movej], &editp[movei][movej]);

                        }

                    }

                    editp[i][ve + 1].ste = 1;

                }

                flag = 1;

            }

            if(flag == 1)

            {

                break;

            }

        }

    }

}

9.16 Boxtrdl.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  树状图删框函数*

*Function list :  deletebox\_tr 删除，将所有的子节点一并删除，并对其余节点的坐标进行调整*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxtrdl.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: deletebox\_tr*

*Function：删除，将所有的子节点一并删除，并对其余节点的坐标进行调整*

*Calls: movebox*

*vanbox*

*Called By: edit*

*Parameter: editp 要画出的导图*

*ho 水平位置*

*ve 竖直位置*

*Return: 无*

*Author: 倪启源*

*Others: 具体思路可参考单向思维导图*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **deletebox\_tr**(**MINDBOX** (\*editp)[VERTREE], int ho, int ve)

{

    int benlie; *//遍历本列所使用的变量*

    int i, j; *//横纵遍历*

    int movei, movej; *//移动框工具变量*

    for(benlie = 0; benlie < ve; benlie ++)

    {

        if(editp[ho][benlie].ste == 1)

        {

            break;

        }

    }

    if(ho == HORI - 1)

    {

        for(movej = ve; movej < VERTREE; movej ++)

        {

**vanbox**(&editp[ho][movej]);

        }

        return;

    }

    for(i = ho + 1; i < HORI; i ++)*//boxa姊妹情况*

    {

        for(j = 0; j < VERTREE; j ++)

        {

            if(editp[i][j].ste == 1)

            {

                break;

            }

        }

        if(j < ve)

        {

            if(benlie == ve)

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[ho][movej]);

                }

                if(i > ho + 1)

                {

                    for(movei = ho + 1; movei < i; movei ++)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movei = i; movei < HORI; movei ++)

                {

                    for(movej = 0; movej < VERTREE; movej ++)

                    {

**movebox**(&editp[ho + movei - i][movej], &editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[ho][movej]);

                }

                if(i > ho + 1)

                {

                    for(movei = ho + 1; movei < i; movei ++)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                    for(movei = i; movei < HORI; movei ++)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**movebox**(&editp[ho + movei - i + 1][movej], &editp[movei][movej]);

                        }

                    }

                }

            }

            break;

        }

        else if(j == ve)

        {

            if(benlie == ve)

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[ho][movej]);

                }

                if(i > ho + 1)

                {

                    for(movei = ho + 1; movei < i; movei ++)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movei = i; movei < HORI; movei ++)

                {

                    for(movej = 0; movej < VERTREE; movej ++)

                    {

**movebox**(&editp[ho + movei - i][movej], &editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**movebox**(&editp[ho][movej], &editp[i][movej]);

                }

                if(i > ho + 1)

                {

                    for(movei = ho + 1; movei < i; movei ++)

                    {

                        for(movej = 0; movej < VERTREE; movej ++)

                        {

**vanbox**(&editp[movei][movej]);

                        }

                    }

                }

                for(movei = i + 1; movei < HORI; movei ++)

                {

                    for(movej = 0; movej < VERTREE; movej ++)

                    {

**movebox**(&editp[ho + movei - i][movej], &editp[movei][movej]);

                    }

                }

            }

            break;

        }

        else if(j < VERTREE && j > ve)

        {

            continue;

        }

        else

        {

            if(benlie == ve)

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[ho][movej]);

                }

                for(movei = ho + 1; movei < HORI; movei ++)

                {

                    for(movej = 0; movej < VERTREE; movej ++)

                    {

**vanbox**(&editp[movei][movej]);

                    }

                }

            }

            else

            {

                for(movej = ve; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[ho][movej]);

                }

                for(movei = ho + 1; movei < HORI; movei ++)

                {

                    for(movej = 0; movej < VERTREE; movej ++)

                    {

**vanbox**(&editp[movei][movej]);

                    }

                }

            }

            break;

        }

    }

    if(i == HORI)

    {

        for(movej = ve; movej < VERTREE; movej ++)

        {

**vanbox**(&editp[ho][movej]);

        }

        if(ho + 1 < HORI)

        {

            for(movei = ho + 1; movei < HORI; movei ++)

            {

                for(movej = 0; movej < VERTREE; movej ++)

                {

**vanbox**(&editp[movei][movej]);

                }

            }

        }

    }

}

9.17 Editfb.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  具体编辑鱼骨图功能及界面*

*Function list :  editfb 编辑鱼骨图功能*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Editfb.h"

#include <graphics.h>

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: editfb*

*Function：编辑鱼骨图界面功能*

*Calls: edit\_screen*

*Called By: operate*

*Parameter: editp*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **editfb**(FISHINFO \*editp)

{

    int e\_state = 1;              *//表示是否在edit页面内 1为仍在 0为跳转    控制页面跳转*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态*

*// 1 完成键 2 新增键 3 删除键 4 保存键*

    int rankbone; *//存储rankbone下标*

    MINPUT minput = {7, 0, ""};

    int find\_flag = 0; *//表示是否找到框*

**clrmous**(MouseX, MouseY);

**delay**(100);

**edit\_screen**();

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(496 + 25 - 1, 248 - 1, 496 + 25 + 42 + 10 + 42 + 1, 272 + 1); *//盖住两个按钮*

**drawbone**(editp);

**mouseinit**();

    while (e\_state == 1)

    {

**newmouse**(&MouseX, &MouseY, &press);

        while (**bioskey**(1))

        {

**getch**();

        }

        if (**mouse\_press**(640 - 42, 0, 640, 24) == 2) *//完成且未点击 未制作点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

            }

        }

        else if (**mouse\_press**(640 - 42, 0, 640, 24) == 1) *//完成且点击*

        {

            e\_state = 0;

**delay**(100);

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 2) *//新增且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 1) *//新增且点击*

        {

            if (find\_flag == 1 && editp -> bonenum < BONEMAX)

            {

**addbone**(editp, rankbone);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawbone**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 2) *//删除且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 1) *//删除且点击*

        {

            if (find\_flag == 1 && editp -> bonenum > 1)

            {

**deletebone**(editp, rankbone);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawbone**(editp);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(0, 24, 496, 480) == 1) *//选要编辑的框*

        {

            find\_flag = **findbone**(editp, &rankbone);

            if (find\_flag == 1)

            {

**clrmous**(MouseX, MouseY);

**delay**(25); *//画图前先清除鼠标，防止干扰*

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(RED);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(248 - 114 + 164 \* (rankbone % 2), 80 + 24 + rankbone \* 40, 248 - 50 + 164 \* (rankbone % 2), 80 + 40 + rankbone \* 40);

**line**(248, 80 + 20 + rankbone \* 40, 248 - 50 + 100 \* (rankbone % 2), 80 + 40 + rankbone \* 40);

**rectangle**(248 - 114 + 164 \* (rankbone % 2), 80 + 24 + rankbone \* 40, 248 - 50 + 164 \* (rankbone % 2), 80 + 40 + rankbone \* 40);

**edit\_yugu**(248 - 114 + 164 \* (rankbone % 2), 80 + 24 + rankbone \* 40, &minput, &((editp->bone)[rankbone]),BLACK);

**strcpy**(minput.str, "");

                minput.cursor = 0;

            }

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawbone**(editp);

**setcolor**(RED);

**setlinestyle**(0, 0, 3);

            if(find\_flag == 1)

            {

**line**(248, 80 + 20 + rankbone \* 40, 248 - 50 + 100 \* (rankbone % 2), 80 + 40 + rankbone \* 40);

**rectangle**(248 - 114 + 164 \* (rankbone % 2), 80 + 24 + rankbone \* 40, 248 - 50 + 164 \* (rankbone % 2), 80 + 40 + rankbone \* 40);

            }

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

                break;

            case 4:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8 + 24 + 8, 496 + 25 + 42, 24 + 8 + 24 + 8 + 24);

            }

        }

    }

}

9.18 Boxfb.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑鱼骨图操作*

*Function list : findbone 寻找鱼骨*

*drawbone 画出当前的鱼骨图*

*deletebone 删去某根骨头*

*addbone 新增某根骨头*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxfb.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: findbone*

*Function：寻找鱼骨*

*Calls: drawbone*

*Called By: editfb*

*Parameter: edifish 受选择的数组*

*pnum 鱼骨位置*

*Author: 倪启源*

*return flag 是否找到框*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **findbone**(FISHINFO \*edifish, int \*pnum)

{

    int i = 0; *//计数变量*

    int flag = 0; *//flag代表是否flag代表是否选中了所点击的区域，若选中了，则变为1，否则为0*

    while(i < edifish -> bonenum)

    {

        if(**mouse\_press**(248 - 114 + 164 \* (i % 2), 80 + 24 + i \* 40, 248 - 50 + 164 \* (i % 2), 80 + 40 + i \* 40) == 1)

        {

            \*pnum = i;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawbone**(edifish);

**setcolor**(RED);

**setlinestyle**(SOLID\_LINE, 0, 3);

**line**(248, 80 + 20 + i \* 40, 248 - 50 + 100 \* (i % 2), 80 + 40 + i \* 40);

**rectangle**(248 - 114 + 164 \* (i % 2), 80 + 24 + i \* 40, 248 - 50 + 164 \* (i % 2), 80 + 40 + i \* 40);

            flag = 1;

            break;

        }

        i ++;

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: drawbone*

*Function：画出当前的鱼骨图*

*Calls: 无*

*Called By: editfb findbone*

*Parameter: editp 要画出的鱼骨图*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **drawbone**(FISHINFO \*edifish)

{

    int i = 0; *//计数变量*

    int headoffish[12] = *//鱼头轮廓*

    {

        248 - 10, 30, 248 - 40, 80, 248 + 40, 80, 248 + 10, 30, 248, 50, 248 - 10, 30

    };

    int tailoffish[8];

    tailoffish[0] = 248;

    tailoffish[2] = 248 - 30;

    tailoffish[4] = 248 + 30;

    tailoffish[6] = 248;

    tailoffish[1] = (edifish -> bonenum) \* 40 + 80;

    tailoffish[3] = (edifish -> bonenum) \* 40 + 80 + 60;

    tailoffish[5] = (edifish -> bonenum) \* 40 + 80 + 60;

    tailoffish[7] = (edifish -> bonenum) \* 40 + 80;

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(LIGHTGRAY);

**fillpoly**(6, headoffish);

**fillpoly**(4, tailoffish);

**line**(248, 80, 248, (edifish -> bonenum) \* 40 + 80);

**setfillstyle**(SOLID\_FILL, WHITE);

**fillellipse**(248, 65, 10, 10);

    while(i < edifish -> bonenum)

    {

**setcolor**(BLACK);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(248 - 114 + 164 \* (i % 2) + 6, 80 + 24 + i \* 40, edifish->bone[i].content);

**setcolor**(LIGHTGRAY);

**line**(248, 80 + 20 + i \* 40, 248 - 50 + 100 \* (i % 2), 80 + 40 + i \* 40);

**line**(248 - 114 + 228 \* (i % 2) ,80 + 40 + i \* 40, 248 - 50 + 100 \* (i % 2), 80 + 40 + i \* 40);

        i ++;

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: deletebone*

*Function：删去某根骨头*

*Calls: 无*

*Called By: editfb*

*Parameter: edifish 鱼骨图*

*pnum 鱼骨位置*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **deletebone**(FISHINFO \*edifish, int pnum)

{

    int i = pnum; *//计数变量*

    while(i < edifish -> bonenum - 1)

    {

**strcpy**(edifish -> bone[i].content, edifish -> bone[i + 1].content);

        i ++;

    }

**strcpy**(edifish -> bone[i].content, "");

    edifish -> bonenum --;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: addbone*

*Function: 新增某根骨头*

*Calls: 无*

*Called By: editfb*

*Parameter: edifish 鱼骨图*

*pnum 鱼骨位置*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **addbone**(FISHINFO \*edifish, int pnum)

{

    int i = edifish -> bonenum;

    while(i > pnum + 1)

    {

**strcpy**(edifish -> bone[i].content, edifish -> bone[i - 1].content);

        i --;

    }

**strcpy**(edifish -> bone[i].content, "");

    edifish -> bonenum ++;

}

9.19 Edidate.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description: 编辑时间轴功能及界面*

*Function list :  edidate 编辑页面功能*

*edidate\_screen 绘制编辑页面*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Edidate.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edidate*

*Function：编辑时间轴界面功能*

*Calls: edidate\_screen*

*finddate*

*drawdate*

*deletedate*

*rank\_time*

*judge\_time*

*Called By: operate*

*Parameter: edidate 待编辑时间轴的信息*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edidate**(DATEINFO \*edidate)

{

    int e\_state = 1;              *//表示是否在edit页面内 1为仍在 0为跳转    控制页面跳转*

    int state = 0, pre\_state = 0; *//表示鼠标状态的相关值（未点击）*

*// 0 无操作状态*

*// 1 完成键 2 新增键 3 删除键 4 保存键*

    int rankdate;                 *//存储rankdate下标*

    ENTER newdateyear = {496 + 25 + 20, 24 + 8 + 24 + 24, 600 + 20, 24 + 8 + 24 + 24 + 20, 4, 0, 0, ""};

    ENTER newdatemonth = {496 + 25 + 20, 24 + 8 + 24 + 24 + 40, 600 + 20, 24 + 8 + 24 + 24 + 60, 2, 0, 0, ""};

    ENTER newdateday = {496 + 25 + 20, 24 + 8 + 24 + 24 + 80, 600 + 20, 24 + 8 + 24 + 24 + 100, 2, 0, 0, ""};

    ENTER newdatehappen = {496 + 8 + 20, 24 + 8 + 24 + 24 + 160 + 100, 496 + 8 + 90 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100, 7, 0, 0, ""};

    DATEHAPPEN tempsave;

    int find\_flag = 0; *//表示是否找到框*

**clrmous**(MouseX, MouseY);

**delay**(100);

**edidate\_screen**();

**drawdate**(edidate);

**mouseinit**();

    while (e\_state == 1)

    {

**newmouse**(&MouseX, &MouseY, &press);

        if (newdateyear.flag != 1 && newdatemonth.flag != 1 && newdateday.flag != 1 && newdatehappen.flag != 1)

        {

            while (**bioskey**(1))

            {

**getch**();

            }

        } *//FIXME*

**input\_info**(525 + 20, 24 + 8 + 24 + 24 + 2, 5, 1, &newdateyear);

**input\_info**(525 + 20, 24 + 8 + 24 + 24 + 40 + 2, 5, 1, &newdatemonth);

**input\_info**(525 + 20, 24 + 8 + 24 + 24 + 80 + 2, 5, 1, &newdateday);

**input\_info**(510 + 20, 24 + 8 + 24 + 24 + 160 + 2 + 100, 5, 1, &newdatehappen);

        if (**mouse\_press**(640 - 42, 0, 640, 24) == 2) *//完成且未点击 未制作点击*

        {

            pre\_state = state;

            state = 1;

            if (pre\_state != 1) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

            }

        }

        else if (**mouse\_press**(640 - 42, 0, 640, 24) == 1) *//完成且点击*

        {

            e\_state = 0;

**delay**(100);

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 2) *//新增且未点击*

        {

            pre\_state = state;

            state = 2;

            if (pre\_state != 2) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24) == 1) *//新增且点击*

        {

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

            tempsave.year = **atoi**(newdateyear.str);

            tempsave.month = **atoi**(newdatemonth.str);

            tempsave.day = **atoi**(newdateday.str);

**strcpy**(tempsave.content, newdatehappen.str);

            if (**judge\_time**(tempsave) && tempsave.content[0] != '\0')

            {

**rank\_time**(edidate, tempsave);

**strcpy**(newdateyear.str, "");

**strcpy**(newdatemonth.str, "");

**strcpy**(newdateday.str, "");

**strcpy**(newdatehappen.str, "");

                newdateyear.cursor = 0;

                newdatemonth.cursor = 0;

                newdateday.cursor = 0;

                newdatehappen.cursor = 0;

**edidate\_screen**();

**drawdate**(edidate);

            }

            else if (**judge\_time**(tempsave) == 0 && (newdateyear.cursor != 0 || newdatemonth.cursor != 0 || newdateday.cursor != 0)) *//输入了日期但不合法*

            {

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 100 + 10 + 20);

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10 + 20);

**puthz**(496 + 25 + 20, 24 + 8 + 24 + 24 + 100 + 10, "日期非法", 16, 16, DARKGRAY);

            }

            else if ((newdateyear.cursor != 0 || newdatemonth.cursor != 0 || newdateday.cursor != 0) && newdatehappen.cursor == 0) *//输入了日期合法文字没输入*

            {

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 100 + 10 + 20);

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10 + 20);

**puthz**(496 + 25 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10, "未输入文字", 16, 16, DARKGRAY);

            }

            else if (!(newdateyear.cursor != 0 || newdatemonth.cursor != 0 || newdateday.cursor != 0) && newdatehappen.cursor != 0) *//没输入日期输了文字*

            {

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 100 + 10 + 20);

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10, 496 + 25 + 20 + 80, 24 + 8 + 24 + 24 + 160 + 20 + 100 + 10 + 20);

**puthz**(496 + 25 + 20, 24 + 8 + 24 + 24 + 100 + 10, "未输入日期", 16, 16, DARKGRAY);

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 2) *//删除且未点击*

        {

            pre\_state = state;

            state = 3;

            if (pre\_state != 3) *//防止被重复标亮*

            {

**clrmous**(MouseX, MouseY);

**delay**(5); *//画图前先清除鼠标，防止干扰*

                MouseS = 1; *//手势鼠标*

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

            }

        }

        else if (**mouse\_press**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24) == 1) *//删除且点击*

        {

            if (find\_flag == 1)

            {

**deletedate**(edidate, rankdate);

**edidate\_screen**();

**drawdate**(edidate);

                find\_flag = 0;

            }

        }

        else if (**mouse\_press**(0, 24, 496, 480) == 1) *//选要编辑的框*

        {

**edidate\_screen**();

**drawdate**(edidate);

            find\_flag = **finddate**(edidate, &rankdate);

        }

        else *//无操作状态*

        {

            pre\_state = state;

            state = 0;

        }

        if (pre\_state != state && pre\_state != 0)

        {

**clrmous**(MouseX, MouseY);

**delay**(5);

            switch (pre\_state)

            {

            case 1:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(640 - 42, 0, 640, 24);

                break;

            case 2:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

                break;

            case 3:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

                break;

            case 4:

                MouseS = 0;

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**rectangle**(496 + 25, 24 + 8 + 24 + 8, 496 + 25 + 42, 24 + 8 + 24 + 8 + 24);

            }

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edidate\_screen*

*Function：绘制编辑界面*

*Calls: 无*

*Called By: edidate*

*Parameter: 无*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edidate\_screen**(void)

{

*//背景*

**cleardevice**();

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 0, 496, 480);

**setfillstyle**(SOLID\_FILL, LIGHTCYAN);

**bar**(496, 0, 640, 480);

**bar**(0, 0, 496, 24);

**setcolor**(WHITE);

**setlinestyle**(0, 0, 3);

**line**(496, 0, 496, 480);

**line**(496, 240, 640, 240);

*//画框*

**setfillstyle**(SOLID\_FILL, LIGHTGRAY);

**bar**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

**puthz**(496 + 25 + 4, 24 + 8 + 4, "新增", 16, 18, WHITE);

**bar**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

**puthz**(496 + 25 + 42 + 10 + 4, 24 + 8 + 4, "删除", 16, 18, WHITE);

**bar**(640 - 42, 0, 640, 24);

**puthz**(640 - 42 + 4, 0 + 4, "完成", 16, 18, LIGHTRED);

**setlinestyle**(0, 0, 1);

**setcolor**(WHITE);

**rectangle**(496 + 25, 24 + 8, 496 + 25 + 42, 24 + 8 + 24);

**rectangle**(496 + 25 + 42 + 10, 24 + 8, 496 + 25 + 42 + 10 + 42, 24 + 8 + 24);

**rectangle**(640 - 42, 0, 640, 24);

*//画输入框*

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24, 600 + 20, 24 + 8 + 24 + 24 + 20);                                 *//年*

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 40, 600 + 20, 24 + 8 + 24 + 24 + 60);                            *//月*

**bar**(496 + 25 + 20, 24 + 8 + 24 + 24 + 80, 600 + 20, 24 + 8 + 24 + 24 + 100);                           *//日*

**bar**(496 + 8 + 20, 24 + 8 + 24 + 24 + 160 + 100, 496 + 8 + 90 + 20, 24 + 8 + 24 + 24 + 160 + 20 + 100); *//输入文字框*

*//画文字框*

**setfillstyle**(SOLID\_FILL, BLUE);

**bar**(496 + 5 + 20, 24 + 8 + 24 + 24, 496 + 25 + 20, 24 + 8 + 24 + 24 + 20); *//年*

**setfillstyle**(SOLID\_FILL, GREEN);

**bar**(496 + 5 + 20, 24 + 8 + 24 + 24 + 40, 496 + 25 + 20, 24 + 8 + 24 + 24 + 60); *//月*

**setfillstyle**(SOLID\_FILL, RED);

**bar**(496 + 5 + 20, 24 + 8 + 24 + 24 + 80, 496 + 25 + 20, 24 + 8 + 24 + 24 + 100); *//日*

**puthz**(496 + 5 + 2 + 20, 24 + 8 + 24 + 24 + 2, "年", 16, 0, YELLOW);

**puthz**(496 + 5 + 2 + 20, 24 + 8 + 24 + 24 + 40 + 2, "月", 16, 0, YELLOW);

**puthz**(496 + 5 + 2 + 20, 24 + 8 + 24 + 24 + 80 + 2, "日", 16, 0, YELLOW);

**puthz**(496 + 8 + 20, 24 + 8 + 24 + 24 + 140 + 100, "内容", 16, 16, BLACK);

}

9.20 Boxdate.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑时间轴操作（找，画，删）函数集*

*Function list : finddate 寻找时间*

*drawdate 画出当前的时间轴*

*deletedate 删去某个时间*

*Attention:*

*Author:  倪启源*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "Boxdate.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: finddate*

*Function：寻找时间*

*Calls: drawdate*

*Called By: edidate*

*Parameter: edidate 受选择的时间轴*

*pdatenum 位置*

*Author: 倪启源*

*return flag 是否找到框*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **finddate**(DATEINFO \*edidate, int \*pdatenum)

{

    int i = 0; *//计数变量*

    int flag = 0; *//flag代表是否选中了所点击的区域*

    while(i < edidate -> daynum)

    {

        switch((i / 5) % 2)

        {

            case 0:

            if(**mouse\_press**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 64) == 1)

            {

                \*pdatenum = i;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawdate**(edidate);

**setcolor**(RED);

**setlinestyle**(SOLID\_LINE, 0, 3);

**rectangle**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 64);

                flag = 1;

                break;

            }

            break;

            case 1:

            if(**mouse\_press**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 64) == 1)

            {

                \*pdatenum = i;

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(0, 24, 496, 480);

**drawdate**(edidate);

**setcolor**(RED);

**setlinestyle**(SOLID\_LINE, 0, 3);

**rectangle**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 64);

                flag = 1;

                break;

            }

            break;

        }

        if(flag == 1)

        {

            break;

        }

        i ++;

    }

    return flag;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: drawdate*

*Function：画出当前的时间轴*

*Calls: 无*

*Called By: edidate finddate*

*Parameter: edidate 要画出的时间轴*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **drawdate**(DATEINFO \*edidate)

{

    int i = 0; *//计数变量*

    char year[5]; *//年暂存*

    char month[3]; *//月暂存*

    char day[3]; *//日暂存*

**strcpy**(year, "");

**strcpy**(month, "");

**strcpy**(day, "");

**setlinestyle**(SOLID\_LINE, 0, 3); *//所有线都是如此*

    while(i < edidate -> daynum) *//画框*

    {

**setcolor**(BLACK);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**itoa**(edidate -> date[i].year, year, 10);

**itoa**(edidate -> date[i].month, month, 10);

**itoa**(edidate -> date[i].day, day, 10);

        switch((i / 5) % 2) *//画框,线*

        {

            case 0:

**setcolor**(BLUE);

**outtextxy**(56 + 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80, year);

**setcolor**(GREEN);

**outtextxy**(56 + 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 16, month);

**setcolor**(RED);

**outtextxy**(56 + 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 32, day);

**setcolor**(BLACK);

**outtextxy**(56 + 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 48, edidate -> date[i].content);

**setcolor**(DARKGRAY);

**rectangle**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 64);

**setcolor**(LIGHTGRAY);

**line**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 16, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 16);

**line**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 32, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 32);

**line**(56 + 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 48, 56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 48);

**setcolor**(BROWN);

            if(i % 5 == 4 && i != edidate -> daynum - 1)

            {

**line**(56 + 80 \* (i % 5) + 32, 48 + 24 + (i / 5) \* 80 + 64, 56 + 80 \* (i % 5) + 32, 48 + 24 + (i / 5) \* 80 + 64 + 16);

            }

            else if(i != edidate -> daynum - 1)

            {

**line**(56 + 80 \* (i % 5) + 64, 48 + 24 + (i / 5) \* 80 + 32, 56 + 80 \* (i % 5) + 64 + 16, 48 + 24 + (i / 5) \* 80 + 32);

            }

            break;

            case 1:

**setcolor**(BLUE);

**outtextxy**(496 - 56 - 64 - 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80, year);

**setcolor**(GREEN);

**outtextxy**(496 - 56 - 64 - 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 16, month);

**setcolor**(RED);

**outtextxy**(496 - 56 - 64 - 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 32, day);

**setcolor**(BLACK);

**outtextxy**(496 - 56 - 64 - 80 \* (i % 5) + 6, 48 + 24 + (i / 5) \* 80 + 48, edidate -> date[i].content);

**setcolor**(DARKGRAY);

**rectangle**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 64);

**setcolor**(LIGHTGRAY);

**line**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 16, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 16);

**line**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 32, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 32);

**line**(496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 48, 496 - 56 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 48);

**setcolor**(BROWN);

            if(i % 5 == 4 && i != edidate -> daynum - 1)

            {

**line**(496 - 56 - 80 \* (i % 5) - 32, 48 + 24 + (i / 5) \* 80 + 64, 496 - 56 - 80 \* (i % 5) - 32, 48 + 24 + (i / 5) \* 80 + 64 + 16);

            }

            else if(i != edidate -> daynum - 1)

            {

**line**(496 - 56 - 64 - 80 \* (i % 5) - 16, 48 + 24 + (i / 5) \* 80 + 32 ,496 - 56 - 64 - 80 \* (i % 5), 48 + 24 + (i / 5) \* 80 + 32);

            }

            break;

        }

**strcpy**(year, "");

**strcpy**(month, "");

**strcpy**(day, "");

        i ++;

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: deletedate*

*Function：删去某个时间*

*Calls: 无*

*Called By: edidate*

*Parameter: edidate 时间轴*

*pdatenum 位置*

*Return: 无*

*Author: 倪启源*

*Others: 无*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **deletedate**(DATEINFO \*edidate, int pdatenum)

{

    int i = pdatenum; *//计数变量*

    while(i < edidate -> daynum - 1)

    {

        edidate -> date[i].year = edidate -> date[i + 1].year;

        edidate -> date[i].month = edidate -> date[i + 1].month;

        edidate -> date[i].day = edidate -> date[i + 1].day;

**strcpy**(edidate -> date[i].content, edidate -> date[i + 1].content);

        i ++;

    }

    edidate -> daynum --;

}

9.21 EditT.c

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Description:  编辑思维导图工具函数*

*Function list : edit\_mind*

*judge\_time*

*edit\_yugu*

*rank\_time*

*Attention:*

*Author:  贾田旺*

*History:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

#include "common.h"

#include "EditT.h"

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edit\_mind*

*Function：思维导图内容文本编辑函数*

*Calls: 无*

*Called By:  Edit*

*Edittr*

*Parameter:  x 输入框左边界*

*y 输入框上边界*

*message 输入导图信息结构体*

*mind 导图信息结构体*

*color 显示字符颜色*

*Return:*

*Author: 贾田旺*

*Others: 框图大小 64\*16*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edit\_mind**(int x, int y, MINPUT \*message, MINDBOX \*mind, int color)

{

    char temp;

    int i;

**clrmous**(MouseX, MouseY);

**delay**(100);

**strcpy**(message->str, mind->content);

    message->cursor = **strlen**(message->str);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**setcolor**(color);

**outtextxy**(x + 6, y, message->str);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);                                                                                         *//FIXME*

**line**(x + 6 + (message->cursor) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (message->cursor) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

    while (1)

    {

*//newmouse(&MouseX, &MouseY, &press);*

        if (**kbhit**())

        {

            temp = **getch**();

            if (temp >= '!' && temp <= '~') *//输入一个字符*

            {

                if (message->cursor < message->length)

                {

                    (message->str)[message->cursor] = temp;

                    (message->str)[message->cursor + 1] = '\0';

                    (message->cursor)++;

                }

                if (message->cursor > 0)

                {

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**line**(x + 6 + (message->cursor - 1) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (message->cursor - 1) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

                }

            }

            else if (temp == '\b') *//删除字符*

            {

                if (message->cursor > 0)

                {

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(RED);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(x, y, x + 64, y + 16);

**rectangle**(x, y, x + 64, y + 16);

                    (message->str)[message->cursor - 1] = '\0';

                    (message->cursor)--;

                }

            }

            else if (temp == 13)

            {

                break;

            }

**setcolor**(color);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(x + 6, y, message->str);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**line**(x + 6 + (message->cursor) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (message->cursor) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

        }

    }

**strcpy**(mind->content, message->str);

    return;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: judge\_time*

*Function：判断传入时间是否正确*

*Calls: 无*

*Called By:  Edidate*

*Parameter: time 存储年月日结构体*

*Return: 格式正确返回1，格式错误返回0*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **judge\_time**(DATEHAPPEN time)

{

    int i;

    int monthdays[13] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

    if (time.year > 3000 || time.year < 1)

        return 0;

    if (time.month > 12 || time.month < 1)

        return 0;

    if (time.month != 2)

    {

        if (time.day > 0 && time.day <= monthdays[time.month])

            return 1;

        else

            return 0;

    }

    else

    {

        if (time.month % 4 == 0 && time.month % 100 || time.month % 400 == 0) *//闰年*

        {

            if (time.day <= monthdays[2] + 1)

                return 1;

            else

                return 0;

        }

        else

        {

            if (time.day <= monthdays[2])

                return 1;

            else

                return 0;

        }

    }

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: rank\_time*

*Function：为时间轴输入的时间进行排序*

*Calls: 无*

*Called By: Editdate*

*Parameter:  timeline 当前时间轴内容*

*time     待插入时间点*

*Return: 1 添加成功 0添加失败*

*Author: 贾田旺*

*Others:*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

int **rank\_time**(DATEINFO \*timeline, DATEHAPPEN time)

{

    int i, j, k, n;

    DATEHAPPEN temp;

    if (timeline->daynum == 0)

    {

        timeline->date[0] = time;

        timeline->daynum++;

        return 1;

    }

    else if (timeline->daynum == MAXDATE)

    {

        return 0;

    }

    for (i = timeline->daynum - 1; i >= 0; i--)

    {

        if (timeline->date[i].year < time.year)

        {

            for (j = timeline->daynum - 1; j > i; j--)

            {

                if (timeline->date[j].year == time.year && timeline->date[j].month < time.month)

                {

                    for (k = timeline->daynum - 1; k > j; k--)

                    {

                        if (timeline->date[k].year == time.year && timeline->date[k].month == time.month && timeline->date[k].day < time.day)

                        {

                            for (n = timeline->daynum; n > k + 1; n--)

                            {

                                timeline->date[n] = timeline->date[n - 1];

                            }

                            timeline->date[n] = time;

                            timeline->daynum++;

                            return 1;

                        }

                    }

                    for (n = timeline->daynum; n > j + 1; n--)

                    {

                        timeline->date[n] = timeline->date[n - 1];

                    }

                    timeline->date[n] = time;

                    timeline->daynum++;

                    return 1;

                }

            }

            for (n = timeline->daynum; n > i + 1; n--)

            {

                timeline->date[n] = timeline->date[n - 1];

            }

            timeline->date[n] = time;

            timeline->daynum++;

            return 1;

        }

    }

    for (n = timeline->daynum; n > 0; n--)

    {

        timeline->date[n] = timeline->date[n - 1];

    }

    timeline->date[n] = time;

    timeline->daynum++;

    return 1;

}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Name: edit\_yugu*

*Function：编辑鱼骨图文本*

*Calls: 无*

*Called By: Editfb*

*Parameter: x 输入框左边界*

*y 输入框上边界*

*message 输入导图信息结构体*

*bone 鱼骨框信息结构体*

*color 显示字符颜色*

*Return: 无*

*Author: 贾田旺*

*Others: 鱼骨为线性，长度64*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

void **edit\_yugu**(int x, int y, MINPUT \*info, FISHBONE \*bone, int color)

{

    char tmp;

    int i;

**clrmous**(MouseX, MouseY);

**delay**(100);

**strcpy**(info->str, bone->content);

    info->cursor = **strlen**(info->str);

**setcolor**(color);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(x + 6, y, info->str);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**line**(x + 6 + (info->cursor) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (info->cursor) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

    while (1)

    {

        if (**kbhit**())

        {

            tmp = **getch**();

            if (tmp >= '!' && tmp <= '~') *//输入一个字符*

            {

                if (info->cursor < info->length)

                {

                    (info->str)[info->cursor] = tmp;

                    (info->str)[info->cursor + 1] = '\0';

                    (info->cursor)++;

                }

                if (info->cursor > 0)

                {

**setcolor**(WHITE);

**setlinestyle**(0, 0, 1);

**line**(x + 6 + (info->cursor - 1) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (info->cursor - 1) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

                }

            }

            else if (tmp == '\b') *//删除字符*

            {

                if (info->cursor > 0)

                {

**setlinestyle**(SOLID\_LINE, 0, 3);

**setcolor**(RED);

**setfillstyle**(SOLID\_FILL, WHITE);

**bar**(x, y, x + 64, y + 16);

**rectangle**(x, y, x + 64, y + 16);

                    (info->str)[info->cursor - 1] = '\0';

                    (info->cursor)--;

                }

            }

            else if (tmp == 13)

            {

                break;

            }

**setcolor**(color);

**settextstyle**(SMALL\_FONT, HORIZ\_DIR, 5);

**outtextxy**(x + 6, y, info->str);

**setcolor**(RED);

**setlinestyle**(0, 0, 1);

**line**(x + 6 + (info->cursor) \* (2 \* 5 - 2) - 1, y + 2, x + 6 + (info->cursor) \* (2 \* 5 - 2) - 1, y + 14); *//光标*

        }

    }

**strcpy**(bone->content, info->str);

    return;

}

9.22 main.c

#include "common.h"

#include "main.h"

void **main**(void)

{

    int page = 1;

    char account\_name[8+1] = ""; *//用户名*

    int graphdriver = VGA;

    int graphmode = VGAHI;

**initgraph**(&graphdriver, &graphmode, "C:\\BORLANDC\\BGI");

**mouseinit**();

    while (1)

    {

        switch (page)

        {

        case 1: *//首页，欢迎界面*

            page = **welcome**();

            break;

        case 2: *//注册界面*

            page = **sign\_up**();

            break;

        case 3: *//登录界面*

            page = **log**(account\_name);

            break;

        case 4: *//重置密码页面*

            page = **reset\_key**();

            break;

        case 5: *//编辑主页面*

            page = **operate**(account\_name);

            break;

        case 6:

            page = **help**();

            break;

        case 0:           *//退出程序*

**closegraph**(); *//关闭画图*

**delay**(100);   *//可解决重启动时鼠标状态不稳定，但有学长反应容易崩溃，慎用*

**exit**(0);      *//退出程序*

        }

    }

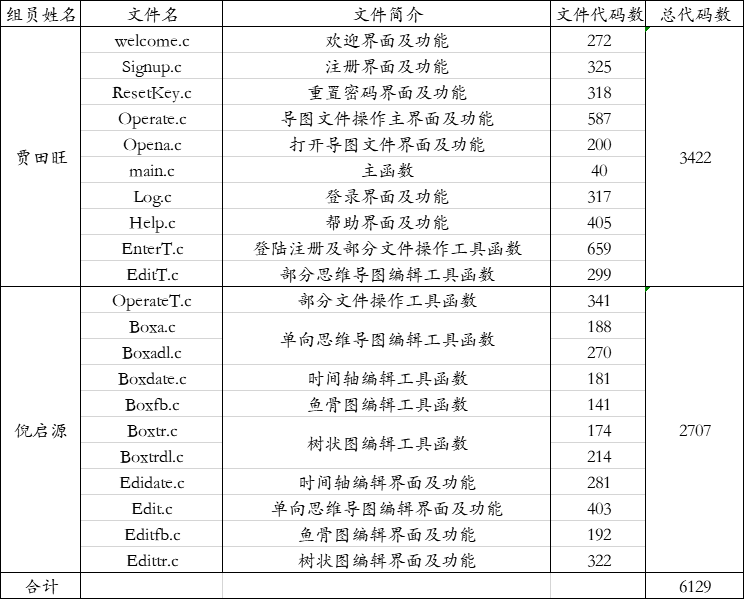
}

10．时间安排及分工

10.1 组员分工

|  |  |
| --- | --- |
| 姓名 | 分工 |
| 贾田旺 | 欢迎界面及功能实现  登录、注册、忘记密码界面及功能实现  打开思维导图文件界面及功能实现  思维导图文件操作总界面及关闭、删除、保存文件功能实现  帮助界面及功能实现 |
| 倪启源 | 新建思维导图文件界面及功能实现  编辑各类思维导图界面及功能实现 |

10.2 组员文件统计与代码量



10.3 时间安排

