浙江大学

程 序 设 计 专 题

大 程 序 报 告

2019~2020 春夏学期 2020 年 6 月 10 日

报告撰写注意事项

1） 图文并茂。文字通顺，语言流畅，无错别字。

2） 书写格式规范，排版良好，内容完整。

3） 存在拼凑、剽窃等现象一律认定为抄袭；0 分

4） 蓝色文字为说明，在最后提交的终稿版本，请删除这些文字。

目 录

疫情数据分析及可视化大程序设计

1 大程序简介

1.1 选题背景及意义

2019 新型冠状病毒（2019-nCoV）于 2019 年末出现于中国湖北省，后于 2020 年初爆发。感染病毒的人会出现程度不同的症状，有的只是发烧或轻微咳嗽，有的会发展为肺炎，有的则更为严重甚至死亡。新型冠状病毒主要的传播途径是呼吸道飞沫传播和接触传播，这种传播方式使其得以在全球范围内广泛传播迅速地开来，对人们的健康及社会经济造成了极大的打击。

社会意义层面，在这种严峻的形势下，开发疫情数据分析及可视化工具，对疫情形势进行分析、以直观形式呈现分析结果对防控疫情具有重大意义。就学习内容而言，借助设计疫情数据分析及可视化工具的大程序，有利于我们综合利用所学知识，强化编程能力，为日后进一步的程序设计以及利用编程解决生活中实际问题打下了坚实的基础。

1.2 目标要求

设计一个疫情数据分析及可视化工具。程序功能包括：1.包括文件操作、编辑、绘图、帮助功能的菜单栏，附带图标工具栏及快捷键；2.即时显示操作中间状态的状态信息栏；3.数据可视化功能；4.疫情预测功能；5.对图表数据的编辑功能。系统程序需采用多文件组织的方式，并且能够实现对疫情数据的保存和读取。

1.3 术语说明

快捷热键：

即快捷键，均采用 Ctrl + 某个字母键 的形式。具体对应关系与可使用的界面。

2 功能需求分析

2.1 业务逻辑架构

2.1.1 表示层

疫情数据分析结果需要借助图形界面直观呈现，通过图形界面进行交互操作。本程序需

要一个完整的图形界面用于结果展示和交互功能。

2.1.2 业务逻辑层

通过鼠标、键盘回调函数接收来自表示层的数据请求，逻辑判断后，向数据访问层提交请求，并传递数据访问结果。

2.1.3 数据访问层

采用链表储存疫情数据和数据名称，进行程序使用过程中的数据读取和存储。最终以 txt 文件储存在用户磁盘，通过文件操作函数进行数据读取。

2.2 功能需求

基于候选题目中提出的基本要求与用户交互体验考虑，梳理本程序功能需求如下。

2.2.1 图表展示

要求基于 libgraphics，对疫情数据进行趋势分析和可视化展示。可通过柱状图、折线图等方式呈现疫情发展趋势和细节数据。最终确定在一个区域的直角坐标系内绘制折线与柱形混合图。使用者可以根据需求放大或缩小，对图表进行查看，自主选择是否显示日期和每个折点对应数据，通过按钮或者直接在 x 轴上拖拽实现移动曲线。选择高亮某条折线。

本程序应当支持同地区的各个处境（例如：感染人数、死亡人数、治愈人数）人数随时间变化数据分析，也支持不同地区的某一特定处境数据随时间变化趋势数据分析与比较。

2.2.2 文件读取和储存

要求能够进行疫情数据保存和读取，实现文件的新建、打开、保存、关闭、退出等操作，从附带文本文件中读取分析所需数据。要求采用多文件组织的方式，分块管理源码文件和头文件，明确程序编写逻辑和功能分区，方便后续维护。

2.2.3 编辑

要求能够进行录入、修改、删除操作。

2.2.4 预测

通过已有数据预测未来一段时间的疫情发展趋势，并在图表中呈现。

2.2.5 菜单栏与快捷键

要求菜单栏具有基本操作，包括文件新建、打开、保存、退出，编辑状态开闭以及帮助。要求实现若干个菜单中常见功能/命令的快捷键。

2.2.6 状态信息栏

窗口底部即时显示操作的中间状态/结果。

2.2.7 引导与提示

程序有一定的使用方法和局限。需要实现弹窗，用作提醒功能。

2.3 数据结构需求

从项目需求角度，链表结构逻辑上连续有序，图形表示过程中可以遍历作图。用户进行放大/缩小图表操作时，双向链表的双指针设置可以做到直接从当前链节回溯，方便操作，运行效率较高。单个元素应当独立，以便实现编辑删改功能。修改文件中储存数据时，链表效率与数组无异，可以全部重新写入，也可以在特定行写入。本程序没有频繁的查找需求。

从内存利用角度，链表数据结构可以实现灵活的内存动态存储。本程序根据外部文件的数据进行展示和操作，动态开辟内存空间可以充分利用内存，提高操作效率。本项目的数据元素个数不确定。链表相比数组方便增加条目，方便数据的不断更新，方便后期维护。

在此基础上，还应当保证数据结构的完整储存，并可以被再次读取。

根据程序的灵活性、编辑操作要求和图像更新要求，双向链表更能胜任疫情数据分析与可视化，作为本程序的数据结构。

2.4 性能需求

2.4.1 可承载数据

疫情发生初始阶段，数据来源是国内各个省份，或者是城市，处理难度在接受范围内。本程序完成时，疫情已经在各个国家爆发开来，处理从各国搜集的数据负荷过大。但目前的程序架构已经可以处理一定数据量。由于图形库精度限制以及窗口设计限制，本组将疫情数据限制在 10 组以内，即来自 10 个地方，天数无具体限制。本程序运行时将提供几组参考数据。

2.4.2 运行效率

本程序在展示之外需要和用户进行有效交互，因此交互功能响应应当较为迅速，尽量低延迟、高刷新。在 simpleGUI 高刷新频率支持下，也应当改良算法，尽量避免遍历操作。

2.4.3 适用平台

目前仅考虑 windows 平台。

2.4.4 编译器要求

应在 visual studio2010 与 Dev C++环境下依给定步骤编译，正常编译运行。

2.4.5 运行要求

应达到基本流畅，正常情况不闪退，能够正常保存和读取文件。

3 程序开发设计

3.1 总体架构设计

3.2 功能模块设计

3.2.1 文件模块

*为方便讲解，以下涉及函数部分均以函数名代替具体声明。

I. 文件操作

1.新建：新建一个新的链表　　fileNew 函数

2.打开：打开列表中的某个文件　　fileOpen 函数

3.保存：保存当前打开的文件　　fileSave 函数

4.退出：退出疫情可视化工具　　exit 函数（自有）

II. 文件编辑与浏览

无需编辑函数，通过直接修改外部附带文本文件储存数据可以方便快捷地绘制出自己想要的数据图表。文件中数据存储的详细内容参见 3.3 数据结构设计。

3.2.2 可视化模块

Ⅰ.界面

1.绘图区：界面中间偏左，大幅界面划分给绘图区。这块界面内，程序利用libgraphics 库函数和本组成员自己编写的逻辑函数，根据文件中的数据绘制图像。这是本程序最重要的部分。

2.按钮： libgraphics 中的 button 函数为本程序的人机交互操作提供了基础。用户可以通过按钮进行编辑、浏览等一系列操作。本程序会对单击按钮的行为进行视觉反馈，并调用相应的回调函数实现功能。

3.菜单：主要用于提供文件操作功能，兼具编辑和帮助功能。

4.状态信息栏：提供程序当前使用状态。

Ⅱ.绘图

本程序每次对图像进行更改后执行 display();函数，内部包括绘图区、按钮等所有可视化函数。绘图区绘制折线图大体步骤如下：

0 确定显示区域头尾节点，地址指针存储在 rpHeadZoom 和 rpTailZoom 中。确定当前绘制折线的序号 rpHead->number[i]

1 根据头尾节点确定显示区域的日期跨度 dateNumber，绘图区 x 轴总长记为 coordinateWidth，计算两个日期之间相隔距离 dx = coordinateWidth * 0.9 / dateNumber;

2 遍历链表获得最高人数 peopleMax 最低人数 0/peopleMin 以及人数差 peopleDelta，记绘图区 y 轴总长为 coordinateHeight，计算单个人数高度 dy = coordinateHeight / peopleDelta;

3 利用 libgraphics 库函数，根据要求作图。主要用到 MovePen 函数和 DrawLine 函数

3.1 折线图

通过 drawFoldLine 函数以及一系列辅助函数作图。

3.2 柱状图

通过 drawBar 函数以及一系列辅助函数作图。

4 横坐标标注算法，人数参考线，人数算法

横坐标依据 dx 一次性遍历打印。人数参考线依据 dy 一次性打印。人数则在每个折点上方标识。

5 图例

统一在折线右端用 drawLabel 函数绘制标签，显示折线对应数据名称。使用自写逻辑函数，避免标签重叠。

Ⅲ.浏览控制

1.左移/右移：

保持比例尺不变，绘图显示整体左移/右移一天。

函数名：buttonLeft　　buttonRight

2.左移/右移到头：

保持比例尺不变，绘图显示左移/右移到头

函数名：buttonLeftest　buttonRightest

3.放大/缩小：

疫情数据繁杂且体量大，需要放大缩小以看得更清晰。放大/缩小操作中显示天数首尾同时收束/扩展一天。

函数名：buttonZoomin buttonZoomout

3.2.3 编辑模块

Ⅰ.按钮编辑

通过点击按钮，达到更改链表的目的。

1.编辑模式 On/Off：

单击按钮，改变全局变量 globalEdit 的值，切换编辑模式。

2.新建日期

在当前打开的文件 RECORD 链表末端增加一个结点，数据由用户输入。

函数名：editNewDate

3.删除最后日期

在当前打开的文件 RECORD 链表末端删除最后一个结点。

函数名：editDeleteLastDate

Ⅱ.图像编辑

通过点击图像，更改全局变量或链表，更改绘图区显示。

1.高亮曲线

单击标签，使选中曲线高亮。

2.高亮折点

3.2.4 预测模块

Ⅰ.预测按钮

单击后计算生成预测链表，接入原 RECORD 链表末端，一并显示在绘图区。

一、 其他

全局变量表

1.int globalActive;全局活跃页面变量 表示当前活跃页面（0123 分别对应主页面/是否框/是否取消框/可输入框等）

2.int globalEdit; 主页面编辑模式变量 表示主页面是否处于编辑模式（0 否 1 是）

3.int globalPredict;主页面预测模式变量 表示主页面是否处于预测模式（0 否 1 是）

4.int globalStatus;全局状态变量，表示当前正在进行的操作

5.int globalFile;全局文件变量，表示当前有无打开的文件（0 无 1 有）

6.char popTip[200];表示在提示框中显示的内容

7.char popInputTip[200];表示在输入框中提示的内容

8.char popInput[200];表示输入框中的用户输入

9.int popStatus2; 表示 prompt2 弹窗状态

10.int popInputStatus = -1;表示 inputBox 弹窗状态（0 修改参数 1 新增线）

3.3 数据结构设计

本组选择的数据结构是双向链表。

3.3.1 链表简介

链表是一种物理存储单元上非连续、非顺序的存储结构，通过链表中的指针链接次序实现数据元素的逻辑顺序。

链表中每一个元素称为结点，链表由一系列结点组成。结点可以在运行时动态生成。每个结点包括两个部分：一个是存储数据元素的数据域，另一个是存储下一个结点地址的指针域。

双向链表是链表的一种，它的每个数据结点中都有两个指针，分别指向直接后继和直接前驱。所以，从双向链表中的任意一个结点开始，都可以很方便地访问它的前驱结点和后继结点。

3.3.2 数据储存文件设计

本小组实验找到的数据源头是一个 python 爬下的今日头条报道数据的仓库。仓库创建者提供了 csv 文件格式和 json 格式，本组选择对 csv 文件数据进行人工筛选。

保留数据用文本文件保存。文件第一行数据包括数据组数，城市名称/人处境代词，日期数第一种是同城市不同处境人数数据，包括年月日，确诊人数、疑似人数、治愈人数和死亡人数。数据间以空格隔开，每条记录末尾换行。

例：2020 年 1 月 26 日某省份的疫情感染新增数据储存为 20200126 3 1 33 1 1 3 2 1 1 14

3.3.3 代码文件内结构设计

代码文件内使用双向链表存储数据。程序运行时通过文件->打开->输入文件名读入目标文件数据，存储在对应链表中。

链表结点定义如下，该定义包含在 linklistMY.h 中：

```
struct record{
    char date[10];                  //日期
    int number[10];                 //各组数据人数
    struct record *prior, *next;    //链表指针
};
typedef struct record RECORD;

struct key{
    char name[20];                  //每组数据字段名
```

```
    struct key *prior, *next;       //链表指针
};
typedef struct key KEY;
```

每个文件将对应两个链表，分别是 RECORD 和 KEY。其中 RECORD 类型用于储存时间信息和各组数据的人数，KEY 类型用于储存每组数据的字段名。

预估用户需求后，做出了以下限制：每个文件最多保存 10 组数据；每组数据的字段名不超过 20 个字符。

3.4 选择双向链表的原因

①能够满足项目需求。

a.链表结构逻辑上连续有序，图形表示时可以遍历作图；

b.用户进行放大/缩小图表操作时，可以通过不同的循环条件控制，链表操作可行。而双向链表的设计，使用户进行缩小图表操作的时候，程序不需要从头遍历图表获取显示的点线集，只需从当前点回溯，提高运行效率。

c.用户编辑数据时，只需要知道具体日期和省份（湖北/浙江/全国）就可以定位到链表的具体链节，通过查找算法找到该链节，修改值，操作思路明确。双向链表的设计为二分查找提供可能性，提高查找效率

d.修改文件中的储存数据效率与数组无异，可以全部重新写入，也可以在特定行写入。

②充分利用计算机内存空间，实现灵活的内存动态管理。

链表不需要事先定义固定长度。根据用户输入的不同的图表大小更改要求，读入到链表中的数据不同，难免有数据长度变化。根据需要动态开辟内存空间，可以节省内存，提高操作效率。

③没有频繁的查找需求。

本项目的数据元素个数不确定。链表具有动态存储分配的数据结构，相比数组方便增加条目，方便数据的不断更新，方便后期维护。

3.4 源代码文件组织设计

3.4.1 文件函数结构

各文件功能简介

文件名	功能简介
common.c	本程序的通用基础函数
common.h	通用基础函数声明
files.c	文件操作函数
files.h	文件操作函数声明
visualization.c	界面刷新函数
	可视化绘图区域功能的所有前后端绘图函数

	可视化有关 button 的后端函数
	可视化功能函数，高亮与拖拽 x 轴移动曲线
	button 绘制函数
	button 排版函数
	菜单栏设置函数
	popwindows 绘制函数
visualization.h	RECORD 和 KEY 结构体
	可视化函数声明
	button 函数声明
	popwindows 绘制函数声明
edit.c	编辑函数
edit.h	编辑函数声明
predict.c	预测函数
predict.h	预测函数声明
linklistMY.c	链表操作函数
linklistMY.h	链表操作函数声明

3.4.2 多文件构成机制

方便起见，将下列文件记为**标准包含**：

```
#include "graphics.h"
#include "extgraph.h"
#include "genlib.h"
#include "simpio.h"
#include "random.h"
#include "strlib.h"
#include "conio.h"
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <math.h>

#include <windows.h>
#include <olectl.h>
#include <mmsystem.h>
#include <wingdi.h>
#include <ole2.h>
#include <ocidl.h>
#include <winuser.h>
```

#include <time.h>
#include "imgui.h"

文件名	包含文件
common.c	#include <stdio.h> #include <stdlib.h> #include <stddef.h> #include <string.h> #include "visualization.h" #include "linklistMY.h"
common.h	#include "linklistMY.h"
files.c	#include <stdio.h> #include <stdlib.h> #include <stddef.h> #include <string.h> #include <time.h> #include "imgui.h" #include "common.h" #include "visualization.h"
file.h	
visualization.c	#include "common.h" #include "linklistMY.h" #include "visualization.h" #include "edit.h" #include "files.h"
visualization.h	标准包含 #include "linklistMY.h"
edit.c	#include "common.h" #include "visualization.h"
edit.h	
predict.c	标准包含 #include "common.h" #include "linklistMY.h" #include "visualization.h" #include "edit.h" #include "files.h"
predict.h	
linklistMY.c	#include <stdlib.h> #include <stdio.h> #include <string.h>

	#include "linklistMY.h"

3.5 函数设计描述

files.c

序号	具体内容
1	函数原型：void fileNew(); 参数描述： 无 返回值描述：无 重要局部变量定义：FILE *fp 重要局部变量用途描述：指向所操作的文件 函数算法描述:无特殊算法,利用了文件操作有关函数,用于新建一个文件(txt)
2	函数原型：void fileOpen(); 参数描述：无 返回值描述：无 重要局部变量定义：FILE *fp 重要局部变量用途描述：指向所操作的文件 函数算法描述：无特殊算法，利用了文件操作有关函数，用于打开根目录下已存在的一个指定文件
3	函数原型：void fileSave(); 参数描述：无 返回值描述：无 重要局部变量定义：FILE *fp 重要局部变量用途描述：指向所操作的文件 函数算法描述：无特殊算法，利用了文件操作有关函数，用于将打开或新建的文件保存

edit.c

序号	具体内容
1	函数原型：void editOnOff(); 参数描述：无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：改变全局变量 globalEdit 的值，使其在 0，1 间切换，1 为 on，0 为 off

2	函数原型：void editNewDate(); 参数描述：无 返回值描述：无 重要局部变量定义： 重要局部变量用途描述： 函数算法描述：利用 linklistMY 中函数，给当前文件的链表新建一个结点
3	函数原型：void editDeleteLastDate(); 参数描述：无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：利用 linklistMY 中函数，删除当前文件的链表的一个结点

common.c

序号	具体内容
1	函数原型：char *nextDate(char today[]); 参数描述： 给定日期 返回值描述：给定日期的下一天 重要局部变量定义：int mAdd 重要局部变量用途描述：判断月份是否进位 函数算法描述：一堆 if 判断条件暴力求出下一天
2	函数原型：void console(char s[]); 参数描述：需要提示用户的信息 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：通过改变全局变量 popTip 的值改变提示框中显示内容；通过改变全局变量 globalActive 的值使得 diplay 函数绘制提示框；

linklistMY.c

序号	具体内容
1	函数原型：RECORD* searchLinkRECORD(RECORD* head, char datex[]); 参数描述： RECORD* head 链表头结点，datex[]需要查找的日期 返回值描述：返回该日期结点指针，未找到则返回 NULL 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：遍历搜寻结点并返回

2	函数原型：void writeLinkRECORD(RECORD* p0,int fieldNum, int numx); 参数描述： RECORD* p0：需要更改的结点；int fieldNum：字段序号； int numx：更改后的最新值 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：更改结点的值
3	函数原型：int getFieldNum(KEY *p0); 参数描述： 给定的 KEY 结点 返回值描述：该 KEY 在链表中的序号 重要局部变量定义：int count 重要局部变量用途描述：统计 KEY 结点之前结点个数 函数算法描述：遍历统计之前结点个数即为序号

visualization.c

序号	具体内容
1	函数原型：void display(); 功能描述：引用各部分可视化函数和弹窗函数，每次刷新界面绘图使用。 参数描述： 无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：纳入清屏函数、绘图函数、绘制 Button 函数、绘制菜单函数，以及全局变量判断 if 语句。
2	函数原型：void drawPic(); 功能描述：调用相关函数，进行画图区域内图像绘制，包括绘图区边框、绘图区坐标系、折线图、柱状图、标注线的名字。 参数描述： 无 返回值描述：无 重要局部变量定义： double sx, sy; double dx, dy; double barDy; int n; int dateNumber; 重要局部变量用途描述： (sx, sy)第一天起笔点坐标； dx 折线绘图每个点之间 x 轴坐标标准间隔； dy 折线绘图每个点之间 y 轴坐标标准间隔； barDy 柱状图每个柱形 y 轴标准间隔；

	n 折线数量； dateNumber 显示区域内天数 函数算法描述：简单计算。

3	函数原型：void drawArea();
	功能描述：绘图区边框绘制，坐标系 x/y 轴绘制
	参数描述： 无
	返回值描述：无
	重要局部变量定义：
	double areaWidth, areaHeight; //长方形 area 的宽、长
	double areaX, areaY; //图像 area 左下角点
	double fontH;
	重要局部变量用途描述：
	areaWidth 绘图区宽度；
	areaHeight 绘图区高度；
	(areaX, areaY)绘图区左下角点坐标，绘图起笔点；
	fontH 字体高度，用于计算标注的“x”和“y”位置
	函数算法描述：调用 libgraphics 库中的函数，绘制长方形绘图区域，绘制坐标轴和坐标轴小箭头，标注 x/y 轴。

4	函数原型：int getDateNumber(RECORD *rpHeadZoom, RECORD *rpTailZoom); 功能描述：获得并返回目前显示数据的天数。 参数描述：两个全局变量，rpHeadZoom 起始日期对应结构地址，rpTailZoom 结束日期对应结构地址 返回值描述：目前显示数据的天数。 重要局部变量定义： int count = 0; RECORD* temp; 重要局部变量用途描述： 存放计算结果，作为返回值； temp 暂时指针，保存指针位置。 函数算法描述：遍历得到 rpHeadZoom 和 rpTailZoom 节点之间的链节数量。
5	函数原型：void drawFoldLine(double sx, double sy, double dx, double dy, int peopleMin, int peopleMax,int dateNumber, int i); 功能描述：绘制折线；人数标明 参数描述： 全局变量 rpHeadZoom 起始日期对应结构地址； 全局变量 rpTailZoom 结束日期对应结构地址； (sx, sy)起笔点坐标； (dx, dy)每个点横纵坐标变化标准间隔； i 现在绘制折线的序号； peopleMin, peopleMax 目前读取文件中数据的人数最小值和最大值； dateNumber 显示区域内天数 返回值描述：无 重要局部变量定义： double nowPointY, lastPointY;

	int j; RECORD* temp; 重要局部变量用途描述： nowPointY 当前点基准线以上 delta 值； lastPointY 上一个点基准线以上 delta 值； j 辅助定位绘图点； temp 暂时指针，保存指针位置。 函数算法描述：调用 judgeHighLightPen(i);确定当前绘制折线是否被高亮。遍历 rpHeadZoom 和 rpTailZoom 之间部分链表，通过当前数据和人数最小值 peopleMin 的差值，结合计算获得当前点 y 坐标，存入 nowPointY，不断更新，调用 libgraphics 库函数绘制折线。每个点可选择标注具体人数数值。存储折线终点坐标进入 highLight[10][4]全局变量，高亮判断用。
6	函数原型：int getKeyNumber(KEY *p); 功能描述：获得总共需要绘制多少条折线。 参数描述：全局 KEY*类型变量，存放数据名字链表的头地址。 返回值描述：n 为需要绘制折线的数量。 重要局部变量定义： int n; KEY *temp; 重要局部变量用途描述： n 存放计算结果，作为返回值。 temp 暂时指针，保存指针位置。 函数算法描述：遍历链表。
7	函数原型：void drawYLine(double sy); 功能描述：绘制 y 轴参考线。 参数描述：绘制起始点 y 轴坐标。 返回值描述：无 重要局部变量定义：double d10, ny; 重要局部变量用途描述： d10 参考线间隔 ny 即 nowY，现在笔所在 y 轴坐标 函数算法描述：for 循环。
8	函数原型：void buttonZoomIn(); 功能描述：“放大”button 的后端函数 参数描述： 全局变量 rpHeadZoom 起始日期对应结构地址； 全局变量 rpTailZoom 结束日期对应结构地址； 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：调用 judgeZoomIn();判断是否可以继续放大，若可以则 rpHeadZoom 往末端移动一个链节，rpTailZoom 往头部移动一个链节。若不可以，则终止操作，并弹窗提示
9	函数原型：int judgeZoomIn();

	功能描述：判断是否可以继续放大。 参数描述： 全局变量 rpHeadZoom 起始日期对应结构地址； 全局变量 rpTailZoom 结束日期对应结构地址； 返回值描述：可以放大返回 1，不可以放大返回 0。 重要局部变量定义：int count; 重要局部变量用途描述：存放计算结果，作为返回值判断依据。 函数算法描述：根据 rpHeadZoom 和 rpTailZoom 之间链节数，计算剩下显示天数，多于 2 天则可以继续放大，少于或等于两天则不能继续放大。
10	函数原型：void buttonZoomOut(); 功能描述：“缩小”button 的后端函数 参数描述： 全局变量 rpHeadZoom 起始日期对应结构地址； 全局变量 rpTailZoom 结束日期对应结构地址； 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：根据 rpHeadZoom 是不是链表头节点、rpTailZoom 是不是链表尾节点以及当前显示数据是否超过 30 天判断是否可以继续缩小，可以则 rpHeadZoom 往头部移动一个链节，rpTailZoom 往尾部移动一个链节。
11	函数原型：int getTotalPeopleNumber(int n, int* peopleMax, int *peopleMin); 功能描述：获得当前文件数据中人数的最大值、最小值，借助指针返回。函数本身返回 max 和 min 的差值，用作 y 轴坐标确定。 参数描述： n 为需要绘制折线的数量； peopleMax 返回值指针，指向人数最大值； peopleMin 返回值指针，指向人数最小值 返回值描述： 本身 int 类型返回 max 和 min 的差值； peopleMax 返回值指针，指向人数最大值； peopleMin 返回值指针，指向人数最小值 重要局部变量定义： int max, min; RECORD* temp; 重要局部变量用途描述： max 和 min 存放计算结果，作为返回值计算依据； temp 暂时指针，保存指针位置。 函数算法描述：遍历链表。
12	函数原型：void drawBar(double sx, double dx, double barDy, int n, int j, RECORD *p); 功能描述：绘制一天数据对应的柱状图。 参数描述： dx 折线绘图每个点之间 x 轴坐标标准间隔；

	barDy 柱状图每个柱形 y 轴标准间隔； n 折线数量； j 是 x 轴辅助定位系数； p 目前这天数据对应所在链节地址 返回值描述：无 重要局部变量定义： double width, x, h; 重要局部变量用途描述： width 每条柱形宽度； x 该组柱形起笔 x 轴坐标； h 每条柱形高度 函数算法描述：调用 libgraphics 库函数绘制长方形。
13	函数原型：void printDateX(double sx, double dx); 功能描述：绘制 x 轴日期坐标，包括竖版文字转换函数。 参数描述： sx 起笔 x 坐标； dx 折线绘图每个点之间 x 轴坐标标准间隔； 返回值描述：无 重要局部变量定义： RECORD* temp; double coordinateLabelY; char dateChange[10][2]; double fontA; 重要局部变量用途描述： temp 暂时指针，保存指针位置； coordinateLabelY 日期起始 y 轴坐标； dateChange[10][2]存储转换后的竖版日期； fontA 字体高度 函数算法描述：遍历链表，每个链节的 date 字符串转换存入二维数组，再调用 libgraphics 函数绘制 label。
14	函数原型：void peopleLabel(RECORD *temp, double dx, double labelX, double labelY, int peopleMin, int peopleMax, int i); 功能描述：标注折点对应人数。 参数描述： temp 暂时指针，保存指针位置，指向正在绘制折点所在链节； dx 折线绘图每个点之间 x 轴坐标标准间隔； (labelX, labelY)人数标签坐标； peopleMin, peopleMax 目前读取文件中数据的人数最小值和最大值； i 现在绘制折线的序号 返回值描述：无 重要局部变量定义： char num[10]; double r = dx * 0.05; double fontA;

	重要局部变量用途描述： num[10]数组暂时存放 itoa 函数转换得到的字符串； r 存储重点记号圈半径； fontA 字体高度，也是 label 高度，控制 y 坐标用 函数算法描述：调用 libgraphics 函数绘制 label 和圆，利用 itoa 函数将 int 类型转换为 char 类型输出。
15	函数原型：void lineName(double px, double py, int i); 功能描述：标注折线名字。 参数描述： (px, py)是当前线 label 左下角坐标； i 现在绘制折线的序号 返回值描述：无 重要局部变量定义： double fontA, dx; KEY *temp; 重要局部变量用途描述： funtA 字体高度； dx 全局变量控制 box 宽度； temp 暂时指针，保存指针位置 函数算法描述：调用 libgraphics 库函数绘制 box。
16	函数原型：void sortLineName(int n); 功能描述：排序标号 tag 的位置，避免重合。 参数描述： n 总共折线数量 返回值描述：无 重要局部变量定义：int i, j; 重要局部变量用途描述：遍历需要。 函数算法描述：比较各条折线终点 y 轴坐标，根据大小调整 highLight[][4]，序号越小 y 坐标越小。
17	函数原型：void adjustLineName(int n); 功能描述：调整折线名称 label 坐标 参数描述： n 总共折线数量 返回值描述：无 重要局部变量定义： double fontA; int mid, findn, middler; 重要局部变量用途描述： fontA 字体高度； mid 中间折线序号； findn 存储当前循环变量对应的名称序号； middler 储存上一循环时循环变量对应的名称序号 函数算法描述：遍历，二分法比较 y 坐标大小关系。如果两个名称 label 绘图区域可能重合，则调整外缘 label 坐标，重新储存入全局 highLight 二维

	数组。
18	函数原型：int findSort(int n, int i); 功能描述：定位目标序号对应的折线名称。 参数描述： n 折线数量； i 是该折线终点 y 轴坐标从下往上排序序号 返回值描述：对应折线名称在 KEY 链表中的链节序号。 重要局部变量定义：int j; 重要局部变量用途描述：循环变量。 函数算法描述：遍历 highLight 全局二维数组。
19	函数原型：void judgeHighLight(double mx, double my); 功能描述：mouseEvent 里判断鼠标是否点击 box 区域，需要高亮 参数描述：(mx, my)鼠标单击坐标 返回值描述：无 重要局部变量定义： double fontA; 重要局部变量用途描述： fontA 字体高度 函数算法描述： 遍历 highLight 二维数组，对比鼠标点击坐标是否在某个 box 的绘图区域。
20	函数原型：void updateHighLight(int i, int n); 功能描述：更新 highLight[][4]二维数组，标注高亮。 参数描述：i 折线序号，n 总共折线数量。 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：遍历数组。
21	函数原型：void buttonLeft(); 功能描述：图像左移，更新显示数据。 参数描述： 无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：rpHeadZoom 和 rpTailZoom 全局变量各往头部移动一链节。
22	函数原型：void buttonRight(); 功能描述：图像右移，更新显示数据。 参数描述： 无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：rpHeadZoom 和 rpTailZoom 全局变量各往尾部移动一链节。
23	函数原型：void buttonRight(); 功能描述：图像右移，更新显示数据。 参数描述：无

	返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：rpHeadZoom 和 rpTailZoom 全局变量各往尾部移动一链节。
24	函数原型：void buttonLeftest(); 功能描述：图像左移到第一天，更新显示数据。 参数描述：无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：调用 getDateNumber();函数得到天数，rpHeadZoom 全局变量取保留的头指针地址 rpHead，先头部移动到底，从头遍历链表 dateNumber 次获得 rpTailZoom。
25	函数原型：void buttonRightest(); 功能描述：图像左移到第一天，更新显示数据。 参数描述：无 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：调用 getDateNumber();函数得到天数，rpTailZoom 全局变量取保留的尾指针地址 rpTail，先头部移动到底，从尾部开始遍历链表 dateNumber 次获得 rpHeadZoom。
26	函数原型：void connnect(RECORD *rp, RECORD *futurep); 功能描述：连接预测链表和原有链表。 参数描述： 全局变量 rp 储存 RECORD 链表头结点； futurep 储存预测链表头结点 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：双向链表两链节连接。
27	函数原型：void seperate(RECORD *rp); 功能描述：拆分预测链表和原有链表。 参数描述：全局变量 rp 储存 RECORD 链表头结点； 返回值描述：无 重要局部变量定义：无 重要局部变量用途描述：无 函数算法描述：双向链表两链节拆分。
28	函数原型：void buttonCustomize(int day1,int month1,int year1,int day2,int month2, int year2); 功能描述：自定义根据用户输入年月日绘制文件中部分图表。 参数描述： day1，month1，year1 共同构成第一个参数 day2，month2，year2 共同构成第二个参数

	返回值描述：无 重要局部变量定义： int date1, date2; int dateNumber; RECORD *temp; 重要局部变量用途描述： date1 和 date2 存储合成参数，用于查询； dateNumber 存储计算所得显示区域内天数 函数算法描述：遍历链表。

4 部署运行和使用说明

4.1 编译安装

4.1.1 Microsoft Visual C++ 2010 Express 下的编译安装

准备材料：

Microsoft Visual C++ 2010 Express，源代码文件夹，测试数据，libgraphics 和 simpleGUI 文件夹。

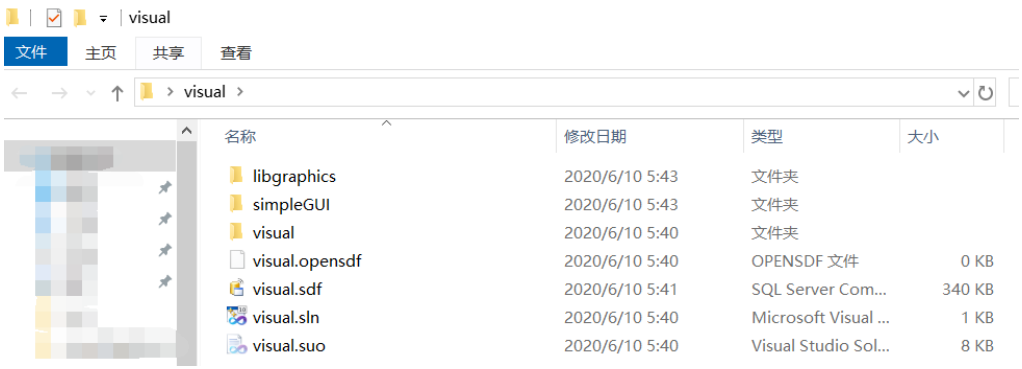
操作过程：

1） 新建项目

菜单栏 File->New->Project，选择 Win32 Project，对工程进行命名，选择保存位置。单击“OK”。弹窗中左侧 Application Settings 里勾选 Empty Project，然后 finish。

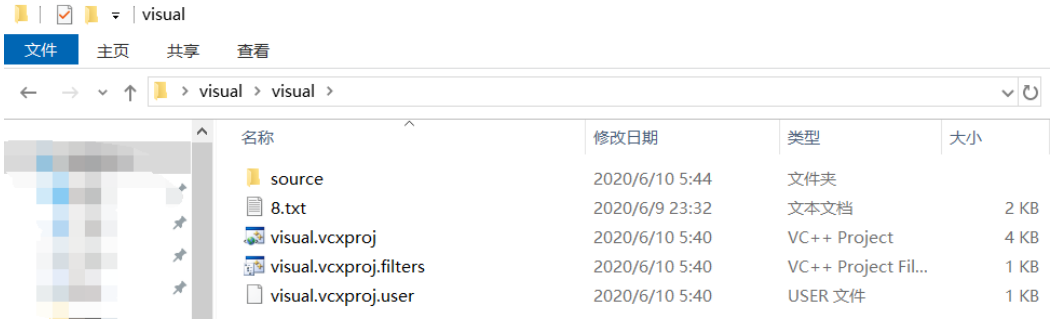
2）导入文件

打开刚刚保存在桌面的仓库 visual 文件夹，将老师的 libgraphics 和 simpleGUI 文件夹复制到此处。



再进入仓库里，也就是当前文件夹下看到的项目文件，及再进入 visual 文件夹。将源代

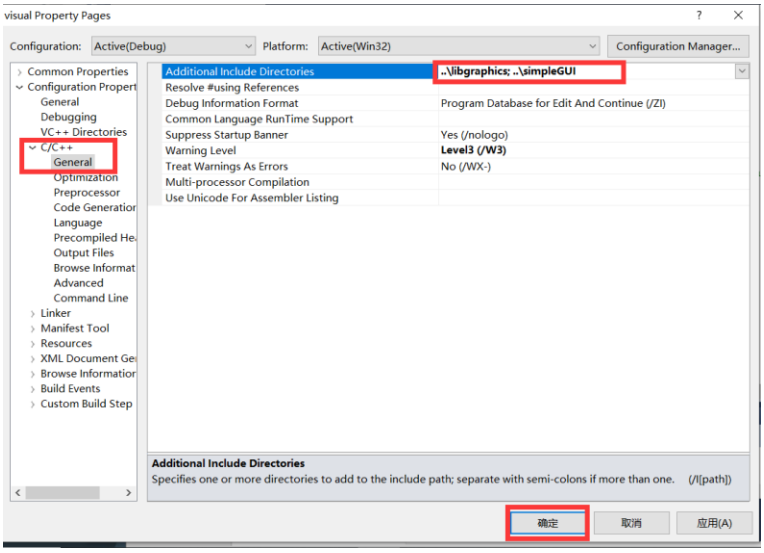
码文件夹和测试数据（例如“8.txt”）复制到这里。



回到 Microsoft Visual C++ 2010 Express 软件界面。右键单击工程->add->new filter，建立新文件夹，命名为 libgraphics&simpleGUI.

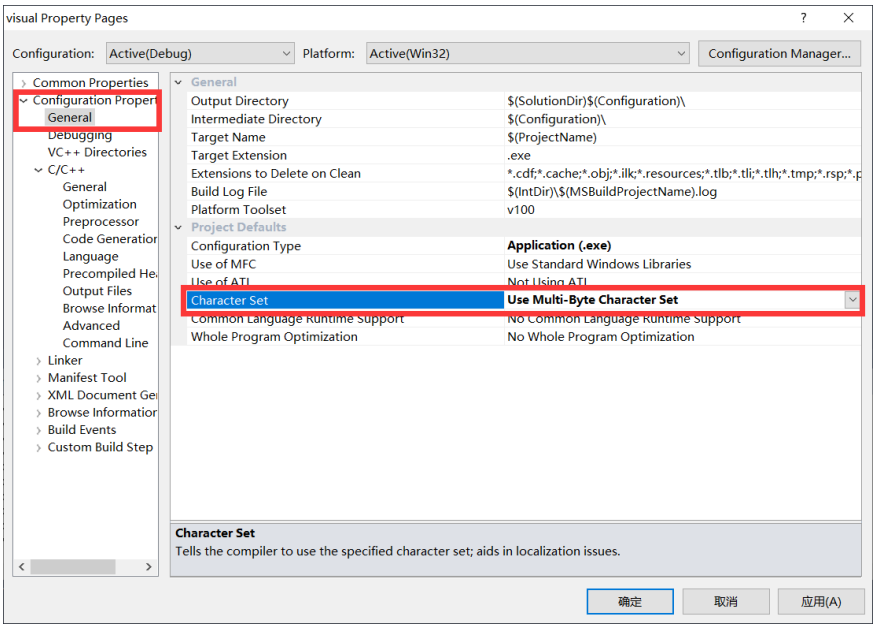
右键单击该新文件夹->add->existing item，选中 libgraphics 文件夹和 simpleGUI 文件夹中内容，导入。同理新建文件夹命名为 codes，导入源代码文件夹内文件。

右键工程文件，选择属性“Properties”。左边菜单栏 C/C++->General，在 Additional include directories 里输入..\libgraphics; ..\simpleGUI，单击确定。



3）修改字符集

右键工程->properties->configuration properties->general->character set 设置为 multi-byte character set。



4）编译运行。

菜单栏 Debug->start debugging 开始编译运行。

4.1.2 Dev C++下的编译安装

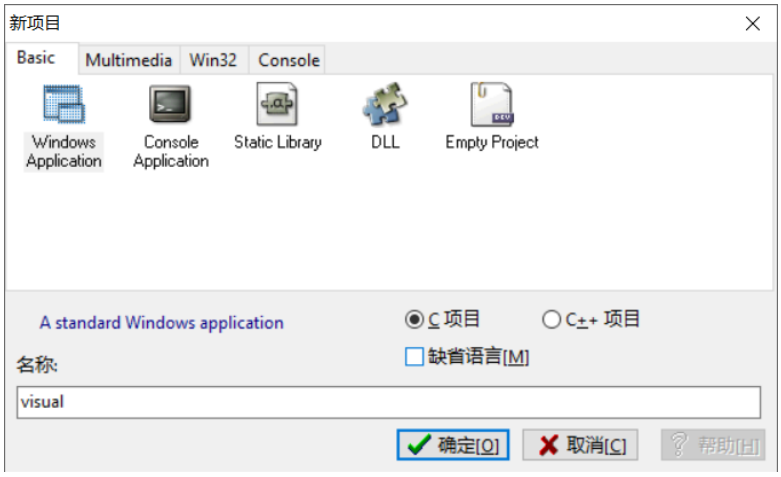
准备材料：

Dev C++，源代码文件夹，测试数据，libgraphics 和 simpleGUI 文件夹。

操作过程：

1） 环境配置

运行 Dev C++，选择文件->新建->项目（这里命名为 visual）。选择“窗体应用程序（Windows Application）”，并勾选“C 项目”。

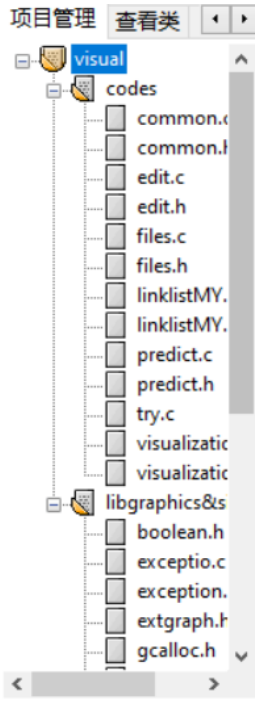


移除预置文件 main.c。

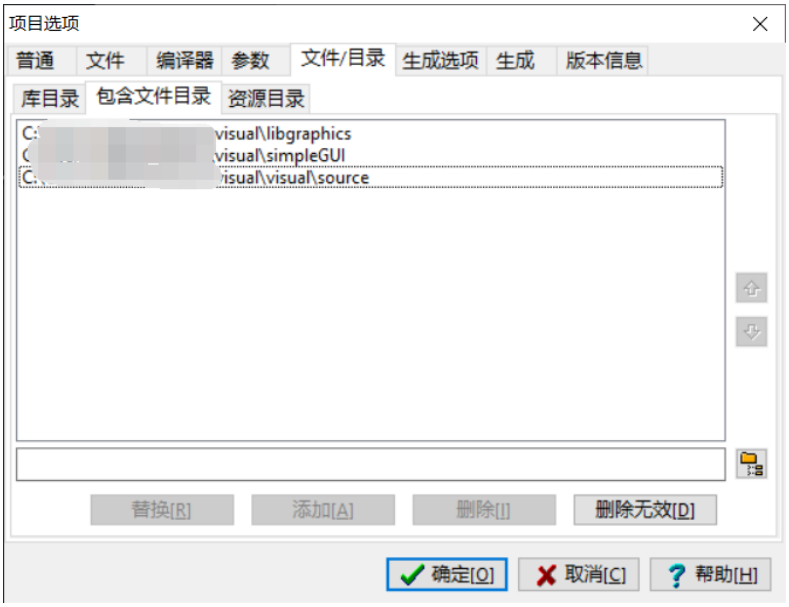
2）添加库函数和源代码

右键单击项目，选择“添加文件夹”，然后添加两个文件夹，分别命名。一个用来存放 libgraphics 库函数和 simpleGUI 库函数，一个用来存源代码。

右键单击文件夹，选择添加，将 libgraphics 和 simpleGUI 内所有文件都添加到文件夹内，同样操作将源代码导入到另一个文件夹下。



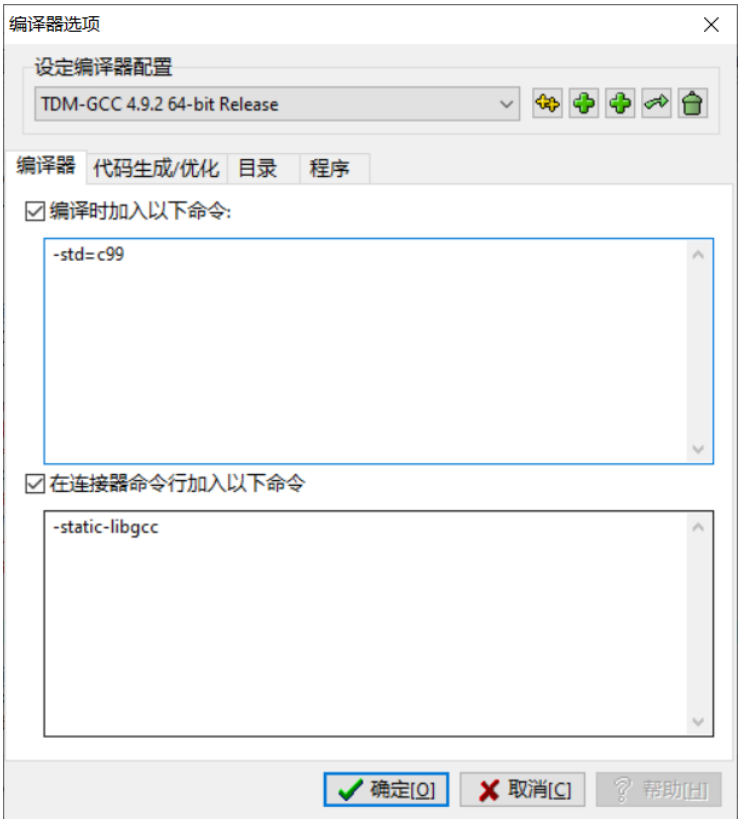
右键单击项目名->项目属性->文件/目录->包含文件目录，依次添加 libgraphics 文件夹、simpleGUI 文件夹和源代码文件夹的路径。可以通过点击右下角橙色的文件图表进行选择并添加。



3）编译器命令

菜单栏->工具->编译选项->编译器中加入命令：-std=c99

然后就可以编译运行了。



4.2 运行测试

测试案例一

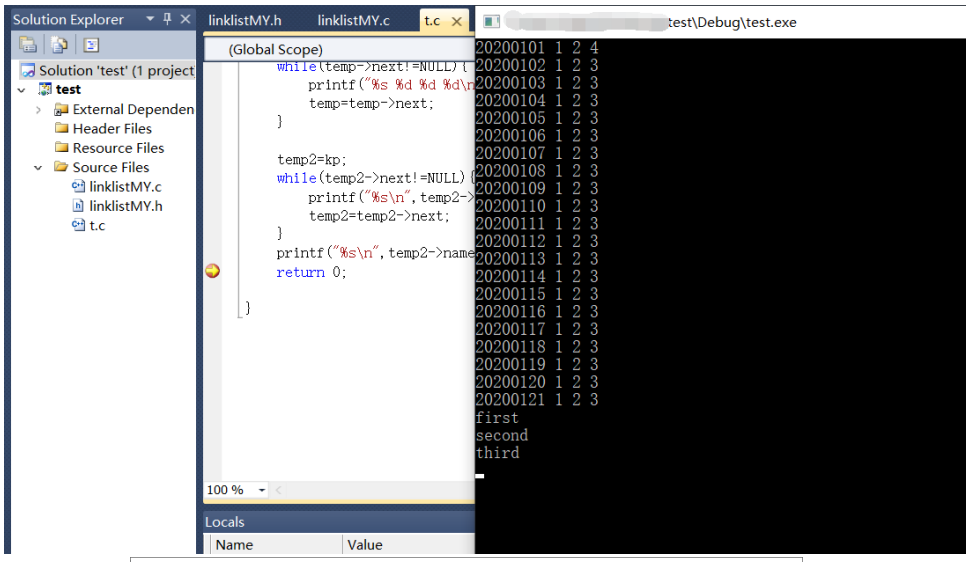
文件：linklist.c; linklistMY.h
错误过程：直接将链表函数接入可视化界面，但并不确定链表函数可用。可以编译，但运行便报错内存溢出，错误定位到 xtoa.c 文件。
测试方法：创建空项目，导入测试文件，接入来自其他文件的链表所需函数，创建 t.c 源码文件写一个基础 main 函数，输入基础数据，利用控制台输出结果与预期比较，校正链表文件错误。再回到小程序工程内部，可以正常运行。
测试数据：

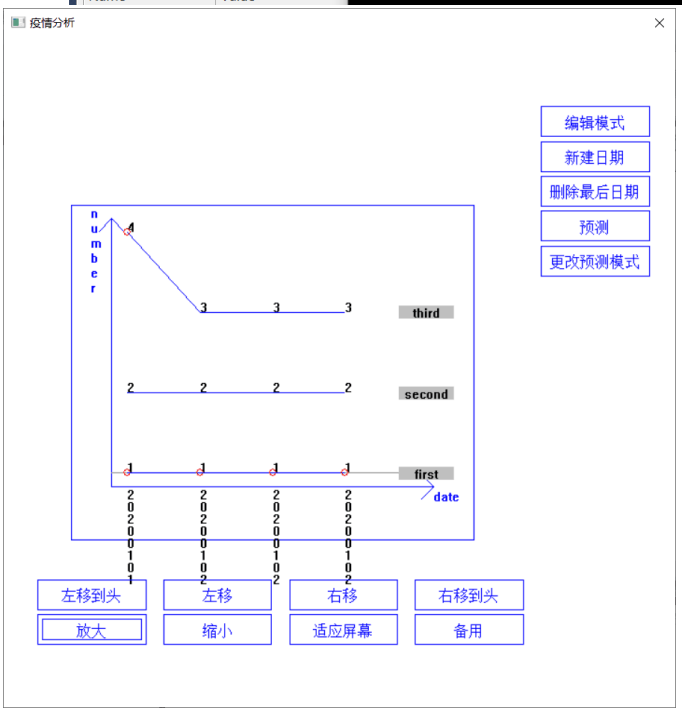
1.RECORD 链表

第一个日期为”20200101”，后续日期为自增。人数数据用 for 循环输入，不进行变化。

2.KEY 链表

开辟数组 char name[4][10]={"first","second","third"};分别读入链节。





附图皆为初期实验结果。

错误原因：链表新增节点函数的参数应该传入指针的指针。

测试代码下附：

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include"linklistMY.h"

RECORD *rp, *rpHeadZoom, *rpTailZoom;
KEY *kp;
//判断是否为闰年函数
int ifRun(int n)
{
```

```
    if ((n % 4 == 0) && (n % 100 != 0) || (n % 400 == 0))
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
//可重复利用的标准函数
//int 类型 1 变 0，0 变 1 函数
void commonTF(int *a){
    if (*a == 0)
    {
        *a=1;
    }
    else{
        *a=0;
    }
}

char *nextDate(char today[])
{
    char year[4];
    char month[2];
    char day[2];    //char 类型年月日
    int y, m, d;    //int 类型年月日
    int mAdd = 0;   //月份是否进位

    //得到 int 类型年月日
    strncpy(year, today, 4); y = atoi(year);
    strncpy(month, today + 4, 2); m = atoi(month);
    strncpy(day, today + 6, 2); d = atoi(day);

    //计算
    if (d == 28 && !ifRun(y) && m == 2)
    {
            d = 0;
            mAdd++;
    }
    else if (d == 29 && ifRun(y) && m == 2)
    {
            d = 0;
            mAdd++;
```

```
    }
    else if (d == 30 && (m == 4 || m == 6 || m == 9 || m == 11))
    {
            d = 0;
            mAdd++;
    }
    else if (d == 31 && m != 2 && m != 4 && m != 6 && m != 9 && m != 11)
    {
            d = 0;
            mAdd++;
    }
    else
    {
        ;
    }
    d = d + 1;
    if (m == 12 && mAdd == 1)
    {
        y++;
        m = 1;
    }
    else{
        m = m + mAdd;
    }
    //年月日拼接成 char 类型
    sprintf(today, "%d%02d%02d", y, m, d);
    return today;
}

int main(void)
{
    RECORD *temp = NULL;
    KEY *temp2 = NULL;
    int i, n;
    //录入数据
    //RECORD
    RECORD *rpHead = NULL,*rpTail = NULL;
    KEY *kpHead = NULL, *kpTail = NULL;
    char datex[10] = "20200101";
    char name[4][10] = {"first", "second", "third"};
    int num[10];

    rpHead = newLinkRECORD();
    rp = rpHead;
```

```
    strcpy(rpHead->date, datex);
    rpHead->number[0] = 1;
    rpHead->number[1] = 2;
    rpHead->number[2] = 4;
    rpTail = rpHead;
    for(i=0; i<=20; i++){
        num[0] = 1;
        num[1] = 2;
        num[2] = 3;
        addLinkRECORD(&rpTail, nextDate(datex), num, 3);
    }
    rpHeadZoom = rp;
    rpTailZoom = rpTail;

    //KEY
    kpHead = newLinkKEY();
    kp = kpHead;
    strcpy(kpHead->name, name[0]);
    kpTail = kpHead;
    addLinkKEY(&kpTail, name[1]);
    addLinkKEY(&kpTail, name[2]);

    temp = rp;
    while (temp->next != NULL)
    {
        printf("%s %d %d %d\n", temp->date, temp->number[0],
               temp->number[1], temp->number[2]);
        temp = temp->next;
    }

    temp2 = kp;
    while (temp2->next != NULL)
    {
        printf("%s\n", temp2->name);
        temp2 = temp2->next;
    }
    printf("%s\n", temp2->name);
    return 0;
}
```

测试案例二

文件：visualization.c; visualization.h

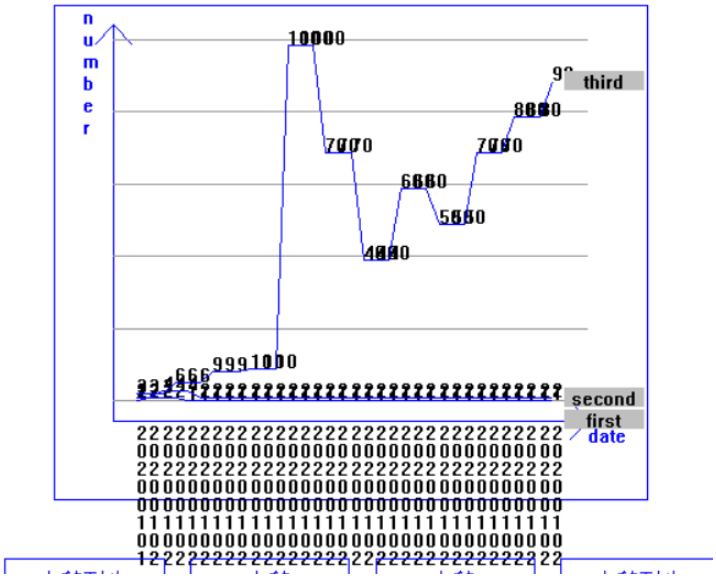
错误过程：未接入文件操作时期，草拟 Main();函数输入部分数据到链表，增加可视化界面绘图折线数量检验极限功能，运行便报错 0xC0000374 堆损坏，错误定位到 xtoa.c 文件。
测试方法：改变导入数据大小不断尝试，以 RECORD 链表为例子检查数据读入代码和链表创建、增加节点操作代码。无果后检查绘图函数 drawFoldLine();
测试数据：Main();函数中假拟一组，利用 for 循环形成递增。
错误原因：测试函数参数错误，导致覆盖写入了不属于动态分配出来的内存，内存溢出。数据量小时没有用到这部分超额内存，不会报错；数据量加大后再申请该部分内存则报错。
测试代码下附：

```
void Main()
{

    RECORD *temp = NULL;
    int i, n;

    录入数据
    RECORD

    char datex[10]="20200101";
    char name[4][10]={"first","second","third"};
    int num[10];

    rpHead=newLinkRECORD();
    rp=rpHead;
    strcpy(rpHead->date,datex);
    rpHead->number[0]=1;
    rpHead->number[1]=2;
    rpHead->number[2]=3;
    rpTail=rpHead;
    for(i=0;i<=10;i++){
        num[0]=1;
        num[1]=2;
        num[2]=3;
        addLinkRECORD(&rpTail,nextDate(datex),num,3);
    }
    rpHeadZoom=rp;
    rpTailZoom=rpTail;

    //录入结束


    //可视化模块开始
    SetWindowTitle("疫情分析");
    SetWindowSize(10, 10); // 单位 - 英寸
    InitGraphics();
```

```
    winWidth=GetWindowWidth();
    winHeight=GetWindowHeight();

    display();
}
```

测试案例三

测试文件：visualization.c 和 visualization.h
错误过程：如图，日期无法正常递增显示。

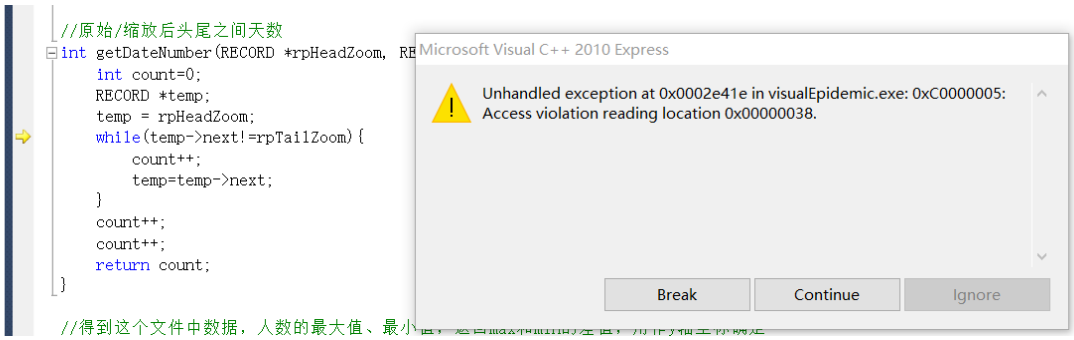


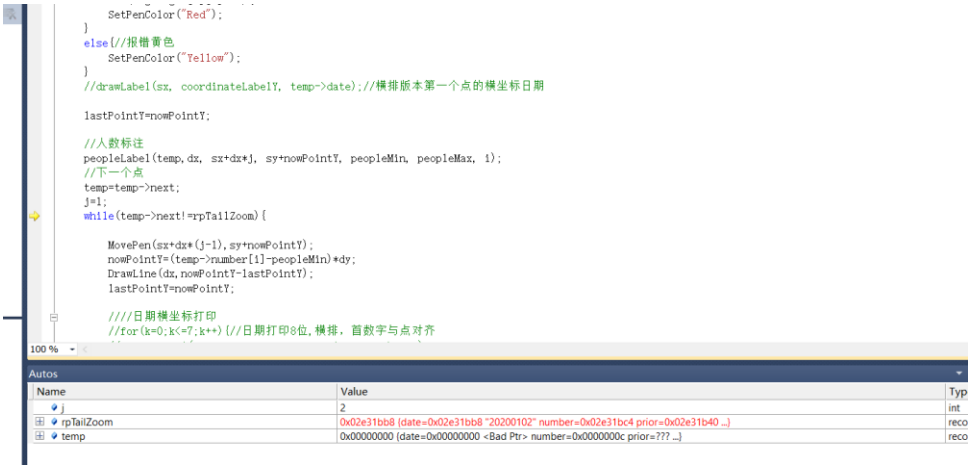
测试方法：改变导入参数，可视化结果没有改变。放大缩小后情况没有变化。于是返回函数寻找原因，定位在 printDateX(double sx, double dx);函数，发现缺少了 temp=temp->next 一步，并且原本写法不符合后期思路，这里不出 bug 也会影响其他功能实现，于是进行了函数重构。修复后日期递增正常。

测试案例四

测试文件：visualization.c 和 visualization.h

错误过程：测试“删除节点”按钮功能时，发现删除到两个节点时报错如下图。
同时引发另一个 bug：链表在有 4 个结点时，先点缩小再点放大，报错相同。初步猜测错误源自初期草拟函数时，限制了放大功能但没有完成程序终止以及弹窗功能，故放大仍然继续进行，与设计思路抵触导致后期程序完成后跑不通。检查完善放大功能后仍没有解决。因而回头检查 drawFoldLine 函数，对绘图函数进行了重构，改变了绘图结束条件。并且对所有涉及极端情况的函数增加了保护措施，改动多处后综合结果解决了问题。





测试方法：同测试案例二
测试数据：同测试案例二

4.3 使用操作

由于 libgraphics 库函数的特殊性质，本程序所有需要输入的部分请全部使用英文，并尽量不要使用 N、O、S、E、T 这五个字母（会触发快捷键）。

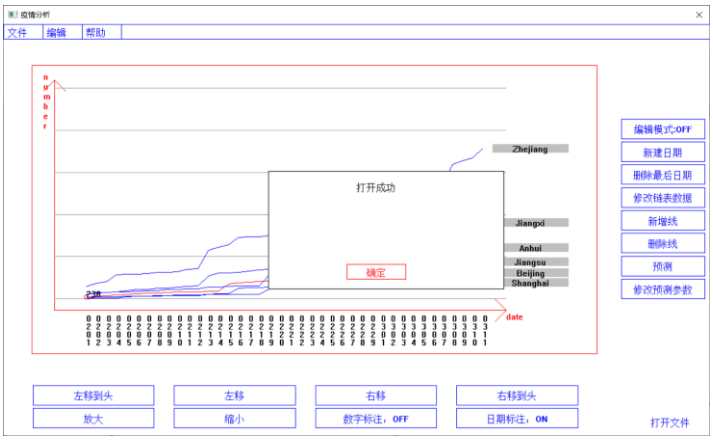
1）基本分区介绍

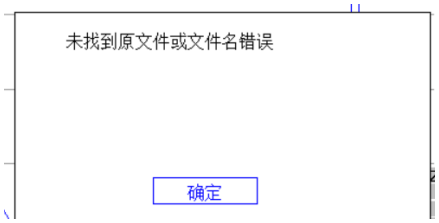
编译运行后初始界面如下，左上角为菜单栏，中间空白部分为待加载的绘图区，右边是编辑模块 button，下方是可视化操作 button，右下角是状态栏。菜单栏具体功能可以自己运行查看，包括新建、打开、保存、退出、编辑和帮助模块。

2） 打开文件

绘图需要数据支持。菜单栏文件->打开，出现弹窗。单击文本框后，输入提供的数据文件名称“6.txt”（本程序提供多个数据文件，择一即可），单击确定。注意输入文件需带后缀名，若发现无法输入点，切换为英文输入法即可。此时可看到状态栏变更为“打开文件”。打开成功后出现弹窗消息“打开成功”。如果输入的文件名非文件夹中有的数据文件，则显示“未找到原文件或文件名错误”。

输入框可以反复删除输入，并且支持光标闪烁。





3） 绘图操作

折线图与柱状图显示：

加载的初始界面如图。绘图区中间是图表，包括折线图和柱状图。若数据文件较大，则初始只显示折线图，放大到一定比例后才显示柱状图。要查看范围外柱状图可以通过移动曲线达成。

图例（标签）：

右边标签意思是城市，依据折线终点位置按顺序对应折线。

“数字标注”按钮的开关控制折线上每个折点的具体数值是否可见。

“日期标注”按钮的开关控制 x 轴日期横坐标是否可见。由于本程序应用于疫情数据分析的特殊背景，可以自行辨认横坐标年份。现在是 2020 年 6 月，>6 月为 2019 年日期，<6 月为 2020 年日期。

放大/缩小操作&左右移动操作：

放大到无法再放大时会弹窗提醒“无法继续放大”。可以通过按钮点击左右移动曲线查看，也可以**在 x 轴上左右拖拽**实现移动。若不想一格格移动，可以使用“左移到头”或“右

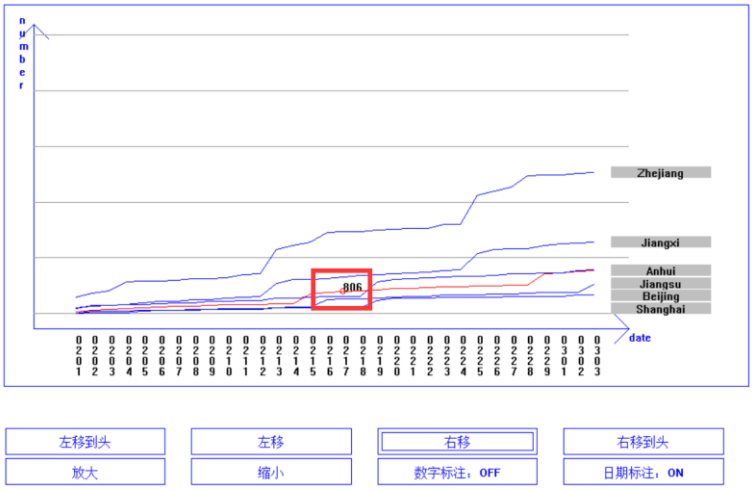
移”到头。

缩小到一定程度后会弹窗提醒“已达到显示结点上限”。这个设置是为了更清晰地看到数据，同时也是防止程序过载。

放大/缩小后标签会随折线末端数据改变，折线和柱状图也会越来越清晰。

光标与高亮：

蓝色线为普通线，红色线为当前所选线。当前所选线即高亮线，线上的小红圈指示左/右移光标所在位置。可以通过鼠标点击图表右侧的折线名称 label 切换高亮线。切换时红圈只转换折线，不会左右偏移，保证了光标作用。光标将在编辑模式中有不可替代的作用。



4） 编辑模式

通过菜单栏->编辑->开启 或者 界面右边的“编辑模式”按钮可以开启编辑模式。若未开启编辑，会弹窗“当前未开启编辑模式”。关闭编辑模式可以防止无意更改数据。

新建文件：菜单栏->文件->新建，或快捷键 Ctrl-N。在弹窗中命名新建数据文件，注意加上后缀“.txt”。新建完成后界面内只有首日期对应点。由于本程序特殊的应用场景，所有数据皆从第一例疫情出现起算，故第一天的日期默认为 2019 年 12 月 01 日。如需更改，需要关闭程序后直接打开数据文件更改。

新建文件后新建日期，注意横坐标变化，已经新建。点击“缩小”按钮可看到新建效果。

新增线：单击界面右边按钮中的“新增线”，在弹窗中输入该折线的命名。通过单击折线名称 label 选中（高亮）该线，即可进行下一步编辑。

删除线：单击界面右边按钮中的“删除线”，删除当前选中线。若图中只剩一条线，将弹窗提醒您无法删除。

新建日期：单击界面右边按钮中的“新建日期”，自动生成下一天日期。

删除日期：删除最后一天数据。注意，需要最右边数据在界面内才能看到变化。当然，删除后再右移到头也可以发现变化。目前可以支持一直删除到最后一个数据。如果要删除最后一个数据，将弹窗提示“链表仅剩一个节点，无法删除”。

修改链表数据：单击界面右边“修改链表数据”按钮，在弹窗中输入数值，即编辑成功。如果是新建文件环境下，需要至少输入 2 天数据才能看到图表。

保存：如果要保存修改，请通过菜单栏->文件->保存。则更改数据会保存入数据文件。所保存的数据文件可以反复读取、关闭。

5 团队合作

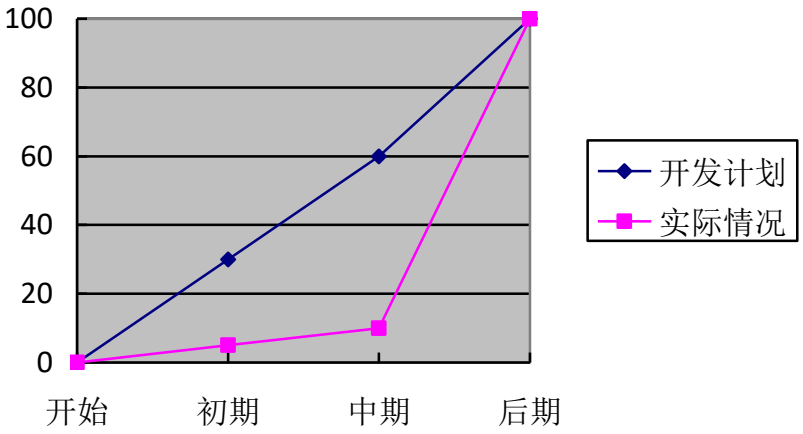
5.1 任务分工

模块	人员	具体内容
文件	A	文件后端函数初稿
	C	文件后端函数重构： 新建文件，打开文件，保存文件
可视化	A	菜单栏初稿
	A	通用函数： 是/否框 是/否/取消框
	B	绘图区实现： 折线图、柱状图、绘图框、xy 轴、横轴日期标注、具体数值标注、折线名称标注
	B	浏览控制按钮后端函数： 放大、缩小、左移、左移到头、右移、右移到头、数字标注、折线高亮
	C	所有按钮前端部分
	C	状态栏
编辑	C	编辑按钮后端函数： 新增/删除线，新增/删除日期，修改链表数据
预测	C	预测链表
基础文件	C+B	链表头文件

5.2 开发计划

人员	时间	内容
A	第一周	链表初稿
B		可视化后端函数初稿，绘图只有折线图
A	第二周	文件、菜单栏初稿
B		重写可用的链表（部分）
C		按钮前端排版
	第三周	链表成稿（C+B）
	第三周	菜单栏函数 debug
B	第三周	visualization.c 文件 debug
	第四周	加入柱形图和标签排版函数
C		编辑模块函数
C		文件函数重构+状态栏
BCA		大报告撰写

Ps：



5.3 编码规范

1.变量命名

变量命名均采用驼峰命名法。

```
//变量示例
double coordinateX;
extern int globalActive;
```

2.h 文件格式

条件编译示例：

```
#ifndef _common_h
#define _common_h
```

```
/*
.h 文件主体
*/
#endif
```

3 预处理指令

```
//引用示例
#include "graphics.h"
```

4. 函数声明/函数定义

左花括号与函数名同行。

```
//函数示例
void empty(){
    ;
}
```

5. 空格的使用

关键字之后要留空格，否则无法辨析关键字。像“if”、“for”、“while”、“catch”等关键字之后应留一个空格再跟左括号“(”，以突出关键字。

逗号“,”之后要留空格，如 Foo(x, y, z)。如果“;”不是一行的结束符号，其后要留空格，如 for (initialization; condition; update)。

函数名后不要留空格，紧跟左括号“(”，以与关键字区别。

一元操作符如“!”、“~”、“++”、“—”、“&”（地址运算符）等与操作数之间不留空格；像“[]”、句点“.”或箭头“->”这类操作符前后不加空格。

赋值操作符、比较操作符、算术操作符、逻辑操作符、位操作符，如“=”、“+=”、“>=”、“<=”、“+”、“*”、“%”、“&&”、“|”、“<<”、“^”等二元操作符的前后应当加空格。

如对于表达式比较长的 for、do、while、switch 语句和 if 语句，为了紧凑起见可以适当地去掉一些空格。

```
//if 语句示例
if (a == b){
    a = -b;
}
```

5.4 合作总结

2020 年 3 月 27 日

会议时间	2020/03/27	参加人员	ABC
会议内容			
会议主题：确定大作业选题为“疫情数据分析与可视化” 主要内容： 1. 选择各个题目的优劣 -主题来源于时事，相比其他几个选题，网络上原创代码少，可出新，发挥空间大。			

-缓解几届以来对老题目的审美疲劳 -可以有很多拓展功能开发 2. 该主题的发展思路 -唯一一个有图例的选题 -滚动条设计 -导入图片做地图 -提供小型文件管理器显示区 -对题目要求的细节探讨 3. 敲定

2020 年 5 月 7 日

会议时间	2020/05/07 2020/05/08	参加人员	ABC
会议内容			
会议主题：开题报告 主要内容： 1.划分部分写初稿 2.细节讨论  重点讨论确定了数据结构的细节，对之后的实际开发工作做了初步设想。 确定了四个模块的方向，对细节实现进行了讨论。			

2020 年 5 月 12 日

会议时间	2020/05/12	参加人员	ABC
会议内容			
会议主题：开题报告互评 主要内容：			

4. 打分 5. 吸取别组设计经验

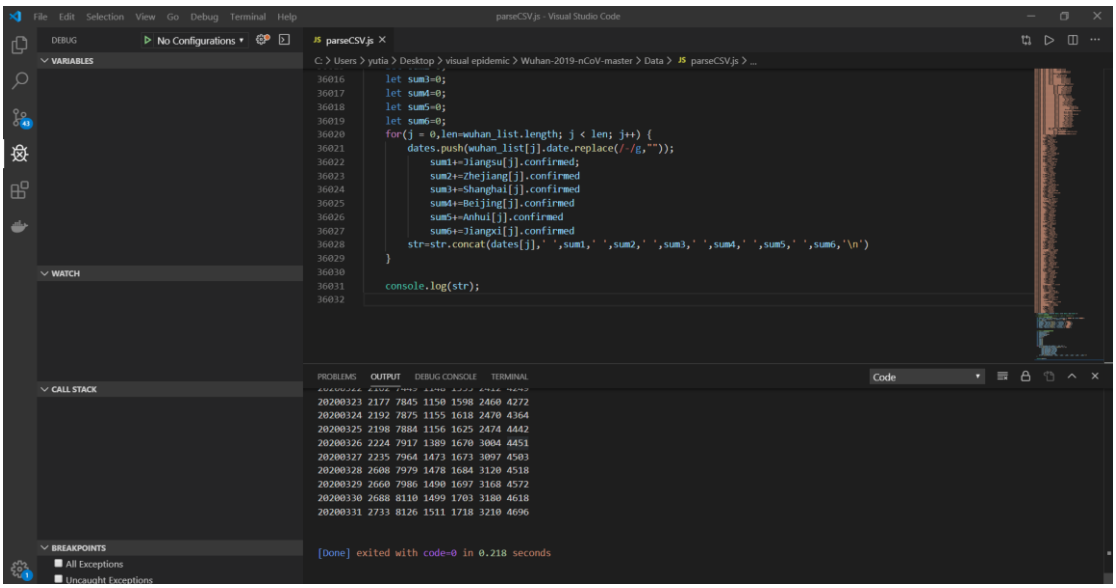
2020 年 5 月 13 日

会议时间	2020/05/13	参加人员	ABC
会议内容			
会议主题：具体任务划分，着手做大作业 主要内容： 1.分任务 因为没有链表函数，后续任务很难进行，故约定当日，最迟一周内做完链表函数。将这部分分给了任务较轻的文件同学。具体任务分配如下： A：文件模块后端函数+可视化模块中的状态栏、菜单栏+通用函数中是/否框，是/否/取消框+链表处理的基础函数 B：可视化模块中的绘图区、浏览控制按钮后端函数+预测模块中预测链表和原链表的连接 C：可视化模块中的所有按钮前端部分+编辑模块编辑按钮后端函数+预测模块中得到预测链表 2.建工程文件 3.细节讨论（链表、图表、编辑功能） 4.将前几天找到的数据从数据源拉下来 5.晚上 9 点汇报当天			

2020 年 6 月 5 日-2020 年 6 月 10 日

会议时间	2020/06/05-2020/06/10	参加人员	BC
会议内容			
会议主题：写功能，合并代码文件，debug 主要内容：所有大作业内容，从代码到大报告			

2020 年 6 月 8 日：难点：获取真实的疫情数据；
解决办法：用 javascript 语言从 github 上一个用 csv 和 json 保存全疫情数据的项目中提取。



5.5 收获感言（注意匿名）

A：

这次的疫情数据分析与可视化工具程序可以说是参与制作的第一个大型的程序，到真正参与这种相对比较大的工程的时候发现这和平时那种作业题还是有很大的区别的，要考虑的东西非常的多。然后在实际操作过程中很多东西也不是能一次性就解决的，可能一个函数就要经过非常多的修改，提出的很多方案也不一定能最好的符合程序，处理各种 bug 更是费了很多工夫，甚至说去借鉴学过的一些例如菜单之类的还是要经过挺多修改才能让它的功能匹配到自己的程序。能够进行这种体验还是有很多收获的，而且团队合作的形式也比较好，从组员那里学到了很多的编程技巧和好思路，有时候自己解决不了的问题求助组员也得到了很好的解决，从而使自己的水平多多少少有点提高吧，是一次有益的活动。

B：

之前尝试模仿老师的 demo 写的几个图形界面经常无法按照我的意愿动起来，挺有挫败感的。这次仍然选择了接下负责可视化界面以及许多后端函数。很早就糊完了初稿，但是由于前期交接不顺利，链表函数很迟才拿到，并且不能正常使用，无法检查。后来还是自己写了适配的链表函数进行 debug 过程。小组合作即使分工下去了，为了自己部分的顺利进行，还是可能要做重合部分的工作的。

这次程序实现还不错，虽然我们呈现的初稿还有一些瑕疵，但是已经能够将想法呈现出来了。我们也通过反复操作探究可能存在的问题。比如删除节点到只剩两个的时候会报错，放大到最后再缩小的时侯也会报错，最后发现是绘图函数跟不上新思路了，需要重构。写程序需要将思路记录下来，后期不断修正。因为时间线拉得长，常常前面的想法和后面不一样了，出现矛盾，拿着后来的思路溯源纠错，花费大把时间。往前想也能发现一开始的思路很

幼稚。最后的几天做了不少这样的重构工作，但是这也让我对我们的疫情数据分析与可视化工具的理解加深了。

最后接入正式的数据呈现出来的结果真的眼前一亮，感到自己终于算是做出个东西了。谢谢坚持修改细节的自己，谢谢找 bug 有一手，修 bug 也有一手的组长。

C：

感觉这次大程作业最困难的部分应该是程序的设计不像之前 pta 上的作业，设计大程序真的是及其困难的一项工作，需要考虑的因素非常之多。本来计划是将程序所有函数接口设计好再开始编程，结果既有很多不合理的函数设计需要删改，又会碰到新的问题需要增添函数，最后实在只能走一步是一步，杂七杂八的函数越来越多。

Debug 和代码合并也是本次作业的两大难点。手动代码合并需要花费很多时间，而每次合并两个成员程序时都会出现很多奇奇怪怪的bug。我们组曾短暂试图使用 git 自带的 merge 功能来进行代码合并，但最终却由于 merge 出现的中文乱码而已失败告终。

这次大程作业让我学到了很多，不光是 c 语言编程的一些技巧。比如在我们获得疫情数据时，需要用到 javaScript 来提取 csv 文件中的信息，而短暂的 git 使用也加强了我们对这一开源的分布式版本控制系统掌握。

6 参考文献资料

由于参考的网络教程或 CSDN 博客实在太多，无法全部记录这里只给出部分文献及网站：

菜鸟教程 https://www.runoob.com/cprogramming/c-tutorial.html

拟合曲线 CSDN 博客 https://blog.csdn.net/beijingmake209/article/details/27565125

编译时提示 conflicting types for 错误的解决办法

https://blog.csdn.net/biubiuibiu/article/details/78326358

业务逻辑详解 https://blog.csdn.net/qq_35038153/article/details/78951206

Github 如何撤销 merge 操作 https://www.jianshu.com/p/6d1d9871ced0

双向链表及创建（C 语言）详解 http://c.biancheng.net/view/3342.html

malloc 申请内存时出现堆已损坏

https://blog.csdn.net/qq_28080659/article/details/80275143

换行符 CR, LF, CR/LF 的区别与关系_git warning

https://www.liupeng.mobi/archives/1622

关于 C++程序编译出现 error C2143: syntax error : missing ';' before 'type'的解决

办法 https://blog.csdn.net/lcr_happy/article/details/52467693

C 如何将二维数组作为返回值

https://blog.csdn.net/wangyang20170901/article/details/79006064

数据结构基础概念篇 https://blog.csdn.net/qq_31196849/article/details/78529724

C 语言读取 csv 文件 https://blog.csdn.net/weixin_40583386/article/details/78540749

c 语言中 char 数组和 int 的转换

https://blog.csdn.net/xiejia32945/article/details/79343581

在 word 中使用 notepad++实现代码的语法高亮

https://www.cnblogs.com/eaglegeek/p/4557994.html

typedef 定义链表 出错

https://blog.csdn.net/ShinSuo/article/details/7312260