

第一部分 算法实现设计说明

1.1 题目

分别以单链表、循环链表、双向链表为例，实现线性表的建立、插入、删除、查找等基本操作。

要求：能够把建立、插入、删除等基本操作的过程随时显示输出来。

1.2 软件功能

对于单链表、循环链表和双向链表，均能实现以下功能：

- (1) 线性表的建立，建立一个带头结点的空链表；
- (2) 线性表插入操作，输入插入索引与插入元素值，将元素插入到链表指定位置；
- (3) 线性表删除操作，输入删除元素的索引，删除该位置的元素；
- (4) 线性表查找操作，输入要查找的元素值，返回第一个等于该值得元素索引；
- (5) 线性表修改操作，输入要修改的索引与修改后的值，修改线性表对应元素；
- (6) 能够对各种特殊情况如索引不合法等进行判断并给出提示。

1.3 设计思想

根据题意，需要对不同链表进行同样的几项基本操作，实现标准交互式图形界面，可以分为算法设计和界面设计两部分。

算法设计方面：对于单链表，本程序链表的建立采用带头结点方式，即每次建立链表都动态申请一个头结点，头结点的后继是 NULL；链表的插入操作需要从头遍历链表，在指定索引处修改插入位置前驱结点的 next 指针指向新结点，新结点的 next 指针则指向该位置原结点；链表的删除操作同样需要从头遍历链表，将指定索引处结点的前驱结点的 next 指针修改为指向该结点的后继结点，并删除当前结点即可；链表的查找操作需要从头遍历链表比较当前访问元素与要查找元素是否相等，若相等则返回索引，若访问完整个链表都找不到相同的元素，则提示查找失败；链表修改操作无需改变链表长度，从头遍历链表找到对应位置元素修改其值即可。循环链表和双向链表的相关操作基本同单链表，不同的是循环链表最后一个结点的 next 指针指向头结点，双向链表除了指向元素后继的 next 指针还有指向元素前驱的 prior 指针。三种链表的实现有所不同，但对于本程序的场景体现不出三种链表的差异。

界面设计方面：可以利用 QT 框架添加按钮、选择框、文本展示框等控件，将三种链表的五项基本操作整合到一个界面中。

1.4 逻辑结构与物理结构

本程序对三种链表进行操作，逻辑结构是顺序结构，物理结构是链式结构，具体结点定义与三种链表的定义如下图所示：

```

struct SNode//单链表结点
{
    int data;
    SNode* next;
};

struct DNode//双向链表结点
{
    int data;
    DNode* prior;
    DNode* next;
};

//单链表
class SingleLinkedList
{
private:
    SNode* head;
    int length;
    Widget* main_widget;

public:
    SingleLinkedList(Widget* widget=NULL);
    void insertAtIndex(int index, int val);
    void deleteAtIndex(int index);
    void modifyAtIndex(int index, int val);
    int searchWithVal(int val);
    void show();
    void destroylinklist();
    ~SingleLinkedList();
};

```

```

//循环链表
class CircleLinkedList
{
private:
    SNode* head;
    int length;
    Widget* main_widget;

public:
    CircleLinkedList(Widget* widget=NULL);
    void insertAtIndex(int index, int val);
    void deleteAtIndex(int index);
    void modifyAtIndex(int index, int val);
    int searchWithVal(int val);
    void show();
    void destroylinklist();
    ~CircleLinkedList();
};

//双向链表
class DoubleLinkedList
{
private:
    DNode* head;
    int length;
    Widget* main_widget;

public:
    DoubleLinkedList(Widget* widget=NULL);
    void insertAtIndex(int index, int val);
    void deleteAtIndex(int index);
    void modifyAtIndex(int index, int val);
    int searchWithVal(int val);
    void show();
    void destroylinklist();
    ~DoubleLinkedList();
};

```

1.5 开发平台

计算机型号：华为 MateBook13

操作系统：Windows 10 家庭中文版

处理器：Intel(R) Core(TM) i7-10510U CPU @ 1.80GHz 2.30 GHz

主体开发语言：C++（支持 C++11）

开发框架：Qt

开发环境：Qt 5.14.2

编辑器: Qt Creator 4.11.1

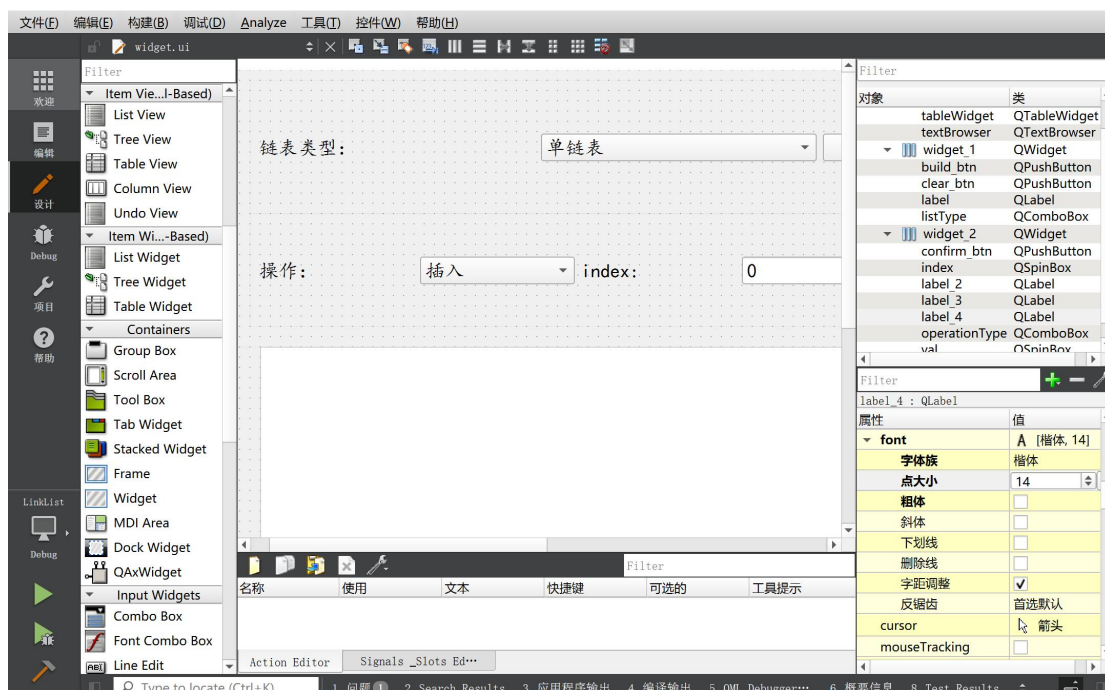
编译器: MinGW 64bit

运行环境: 代码版本使用上述集成环境可正常编译运行, 使用 windeployqt 整合生成的可执行文件版本可在 windows 环境的机型下正常运行。

1.6 系统的运行结果分析说明

1.6.1 调试及开发过程

本次链表操作演示系统的开发采用的是新技术框架 Qt, 同时也是跨平台的, 在 Qt Creator 中开发调试。由于需要设计标准交互式图形界面, 本次软件开发先在 ui 界面放置好相应控件, 将三种链表的五种基本操作整合到一个界面中如下图所示:



在代码中调整好控件位置和样式后, 利用信号与槽的机制连接核心代码与 ui 界面部分。由于是第一次尝试用 Qt 做 ui 界面, 开发调试过程中也遇到了很多问题, 利用头文件 <QDebug> 中的 qDebug() 函数可以将指定信息输出到控制台进行检查。

1.6.2 开发软件达到的成果

(1) 正确性。经过多次调试, 链表操作演示系统达到了预期效果, 界面的运行效果如下图所示, 页面下方左侧表格实时展示链表内容, 右侧表格展示操作记录:

链表操作演示

链表类型:

单链表

创建

清空

操作:

插入

index:

0

val:

0

确定

- (2) 稳定性。程序能够在不同情况下稳定运行，未发现会导致系统崩溃的错误。
- (3) 容错能力。程序具有较好的容错能力，对于不正确的索引能够给出提示，在不同的操作阶段设置不同的模块不可编辑，防止用户误操作造成程序崩溃。如链表未创建时只能选择链表类型并点击创建链表，其余按钮点击无效；而在对链表进行基本操作的过程中则不能点击“创建”按钮，必须清空当前链表释放内存后才可以重新创建链表。

链表操作演示

链表类型:

单链表

创建

清空

操作:

插入

index:

4

val:

1

确定

head	0	0	1

当前列表中的内容: 空

在0处插入元素值为0的结点成功!

当前列表中的内容: 0

正在索引第0项

在1处插入元素值为0的结点成功!

当前列表中的内容: 0 0

正在索引第0项

正在索引第1项

在2处插入元素值为1的结点成功!

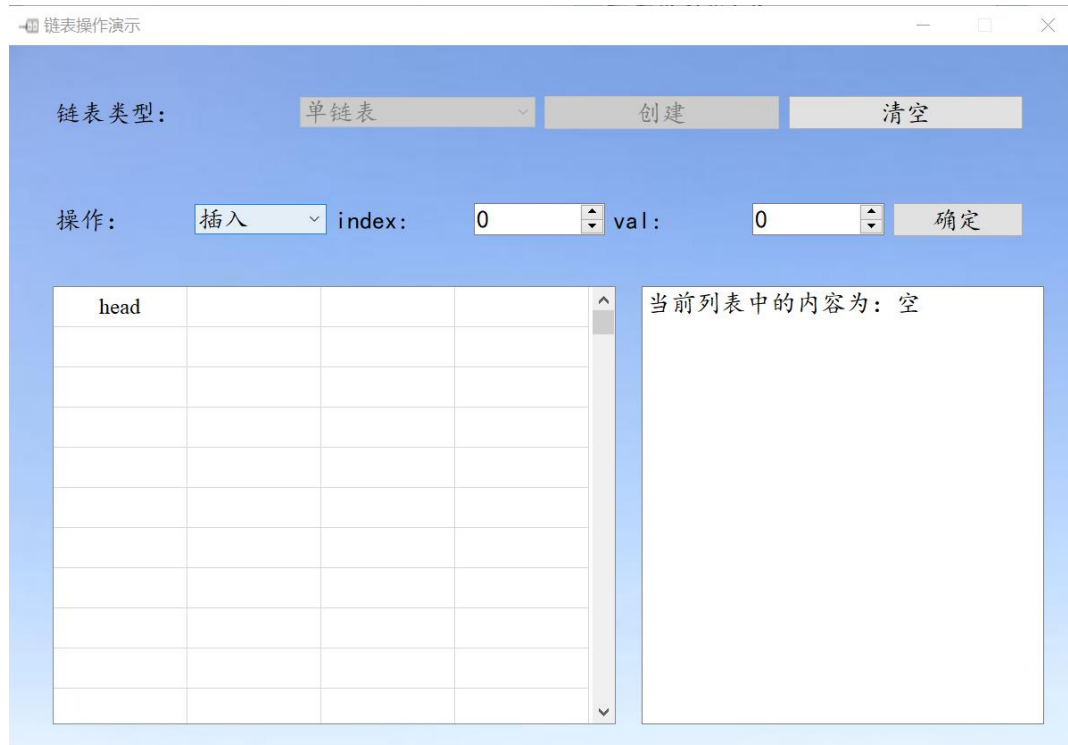
当前列表中的内容: 0 0 1

索引不正确, 插入结点失败!

1.6.3 运行结果分析

尽管单链表、循环链表和双向链表在结构和操作上有些许差异，但对于本链表操作演示系统中，从外观上看并没有分别，因此下面仅分析单链表的运行结果，循环链表和双向链表已验证正确，在此不进行赘述。

首先选择单链表，点击创建，此时链表为只有一个头结点的空链表，如下图所示：



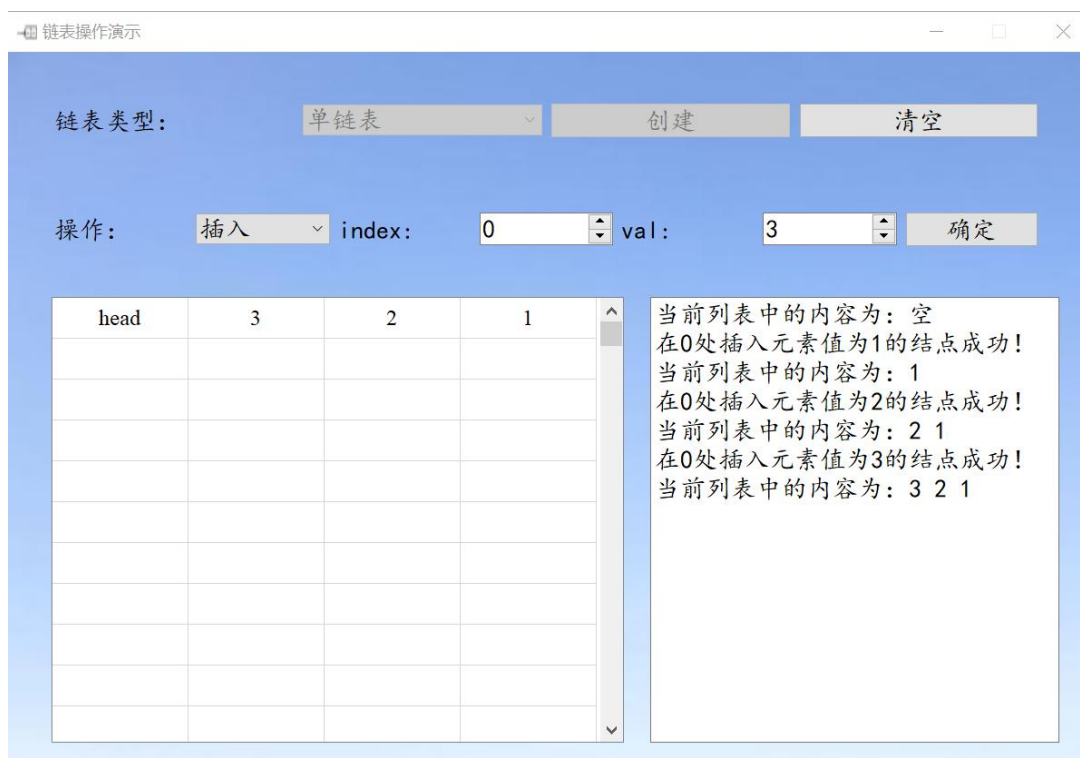
链表类型: 单链表 创建 清空

操作: 插入 index: 0 val: 0 确定

head			

当前列表中的内容为: 空

选择插入，可以向链表中指定索引处插入数据，下图为在链表头依次插入 1, 2, 3 后的结果：



链表类型: 单链表 创建 清空

操作: 插入 index: 0 val: 3 确定

head	3	2	1

当前列表中的内容为: 空
在0处插入元素值为1的结点成功!
当前列表中的内容为: 1
在0处插入元素值为2的结点成功!
当前列表中的内容为: 2 1
在0处插入元素值为3的结点成功!
当前列表中的内容为: 3 2 1

接下来选择删除，删除索引 1 处的元素 2，结果如下图所示：

链表操作演示

链表类型: 单链表 创建 清空

操作: 删除 index: 1 val: 0 确定

head	3	1	

当前列表中的内容为: 空
在0处插入元素值为1的结点成功!
当前列表中的内容为: 1
在0处插入元素值为2的结点成功!
当前列表中的内容为: 2 1
在0处插入元素值为3的结点成功!
当前列表中的内容为: 3 2 1
正在索引第0项
删除索引1处的结点成功!
当前列表中的内容为: 3 1

选择修改，将索引 0 处的元素值修改为 2，结果如下图所示：

链表操作演示

链表类型: 单链表 创建 清空

操作: 修改 index: 1 val: 2 确定

head	3	2	

当前列表中的内容为: 空
在0处插入元素值为1的结点成功!
当前列表中的内容为: 1
在0处插入元素值为2的结点成功!
当前列表中的内容为: 2 1
在0处插入元素值为3的结点成功!
当前列表中的内容为: 3 2 1
正在索引第0项
删除索引1处的结点成功!
当前列表中的内容为: 3 1
正在索引第0项
修改索引1处结点的元素值为2成功!
当前列表中的内容为: 3 2

最后选择查找，查找元素值为 2 的元素，提示索引为 1，如下图所示：

链表操作演示

链表类型：

单链表

创建

清空

操作：

查找

index：

1

val：

2

确定

head	3	2	

当前列表中的内容为：空

在0处插入元素值为1的结点成功！

当前列表中的内容为：1

在0处插入元素值为2的结点成功！

当前列表中的内容为：2 1

在0处插入元素值为3的结点成功！

当前列表中的内容为：3 2 1

正在索引第0项

删除索引1处的结点成功！

当前列表中的内容为：3 1

正在索引第0项

修改索引1处结点的元素值为2成功！

当前列表中的内容为：3 2

值为2对应元素索引为1

点击“清空”按钮可以清空当前链表，回到初始界面：

链表操作演示

链表类型：

单链表

创建

清空

操作：

查找

index：

0

val：

0

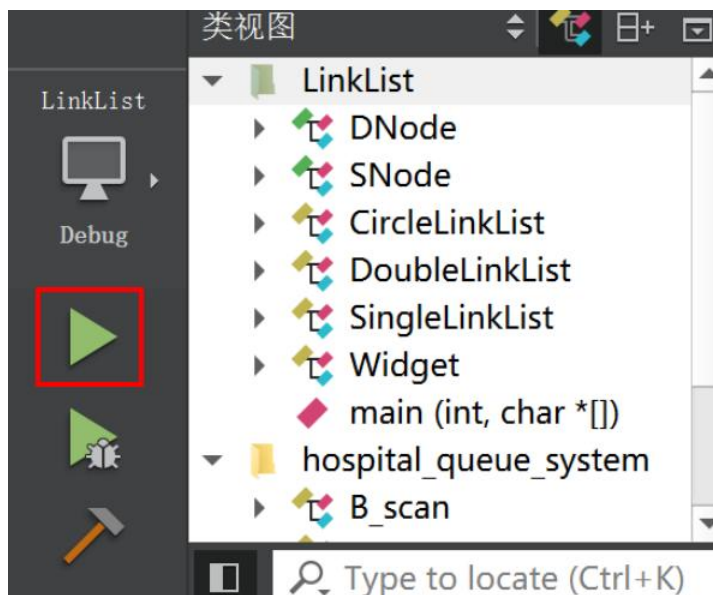
确定

1.7 操作说明

- (1) 双击项目文件打开项目

img	2022/8/28 17:40	文件夹	
circlelinklist.cpp	2022/8/29 13:43	C++ Source	5 KB
circlelinklist.h	2022/8/29 12:39	C/C++ Header	1 KB
doublelinklist.cpp	2022/8/29 13:43	C++ Source	5 KB
doublelinklist.h	2022/8/29 12:39	C/C++ Header	1 KB
LinkedList.pro	2022/8/28 21:06	Qt Project file	2 KB
LinkedList.pro.user	2022/8/28 21:16	Per-User Project O...	23 KB
main.cpp	2022/8/28 20:56	C++ Source	1 KB
Node.h	2022/8/24 14:43	C/C++ Header	1 KB
rsc.qrc	2022/8/28 17:40	QRC 文件	1 KB
singlelinklist.cpp	2022/8/29 13:41	C++ Source	5 KB
singlelinklist.h	2022/8/29 12:39	C/C++ Header	1 KB
widget.cpp	2022/8/29 13:45	C++ Source	7 KB
widget.h	2022/8/29 13:24	C/C++ Header	2 KB
widget.ui	2022/8/29 13:35	Qt UI file	6 KB

(2) 在 Qt Creator 中运行项目



(3) 后续操作同 1.6.3 节，在此不再进行赘述。

(4) 前两步也可以用直接双击 .exe 文件代替，如下图所示：

iconengines	2022/8/29 15:15	文件夹	
imageformats	2022/8/29 15:15	文件夹	
platforms	2022/8/29 15:15	文件夹	
styles	2022/8/29 15:15	文件夹	
translations	2022/8/29 15:15	文件夹	
circlelinklist.o	2022/8/29 13:43	O 文件	495 KB
D3Dcompiler_47.dll	2014/3/11 18:54	应用程序扩展	4,077 KB
doublelinklist.o	2022/8/29 13:43	O 文件	495 KB
libEGL.dll	2020/3/28 3:12	应用程序扩展	67 KB
libgcc_s_seh-1.dll	2018/3/19 23:14	应用程序扩展	73 KB
libGLSv2.dll	2020/3/28 3:12	应用程序扩展	6,214 KB
libstdc++-6.dll	2018/3/19 23:14	应用程序扩展	1,393 KB
libwinpthread-1.dll	2018/3/19 23:14	应用程序扩展	51 KB
LinkedList.exe	2022/8/29 13:45	应用程序	3,149 KB