

****

本科毕业设计(论文)

GRADUATION DESIGN (THESIS)

题 目\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学生姓名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

指导教师\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学 院\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

专业班级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

本科生院制

2015年6月

通信专业离散数学动态演示系统的设计与实现

摘要

离散数学(Discrete mathematics)是研究离散量的结构及其相互关系的数学学科，是现代数学的一个重要分支。它在各个学科领域，特别是在计算机科学领域有着广泛的应用。离散数学作为许多计算机相关专业的专业课程，是程序设计语言、 数据结构、操作系统、编译原理、数字电路、人工智能、数据库、算法等必不可少的先修课程。

对于通信工程专业而言，离散数学也有着重要的地位。比如数字电路，数字信号处 理，信息论与编码等都会有所涉及。但是离散数学是一门相对抽象，对逻辑思维要求很高的学科，大部分学生在学习的过程中会感觉比较吃力。

目前，计算机编程技术相对成熟，各种编程语言和开发环境层出不穷。通过广泛查阅各种辅助教学软件的设计思想，构建一个离散数学课程的动态演示系统，帮助学生理解典型的离散数学知识点，直观的掌握离散数学的部分知识。

**关键词：** 通信工程 离散数学 动态演示系统

**Design and implementation of a dynamic demonstration system of discrete mathematics**

**ABSTRACT**

Discrete mathematics is a subject which discusses discrete structures and their relations, and it is an important branch of the modern mathematics. It is widely used in various fields, especially in the field of computer science. Discrete mathematics is a specialized courses of many computer related subjects, which is an Advanced Placement class for programming language, data structure, operating system, compiler theory, digital circuit, artificial intelligence, database, and algorithm, etc.

As for communication engineering, discrete mathematics also plays an important role. Such as digital circuits, digital signal processing, information theory and coding, will be involved. However, discrete mathematics is relatively abstract and needs high logical thinking, the majority of students in the learning process will feel much difficult.

Currently, computer programming technology is mature, a variety of programming languages and development environments are endless. With widely accessible through a series of auxiliary teaching software design, to build a dynamic discrete mathematics presentation system to help students understand the typical discrete mathematics knowledge, and intuitively grasp of part of the knowledge of discrete mathematics.

目录

1. 绪论
   1. 项目研究背景
   2. 国内外研究现状
   3. 任务概述
   4. 论文的结构
2. 系统概要设计
   1. 面向对象的编程思想
      1. 面向对象简介
      2. C++ 简介
      3. Qt 简介
      4. Git 简介
   2. 系统设计
      1. 系统的总体结构
      2. 系统的演示结构
   3. 系统界面设计
   4. 系统基本功能
   5. 系统主要模块
      1. 主窗口
      2. 声明窗口
      3. 帮助窗口
      4. 功能模块
3. 功能模块分析与实现
   1. 验证表达式是否为命题公式
   2. 求任意公式的真值表
   3. 求任意两个集合的运算
   4. 判断关系的性质
4. 系统演示
   1. 系统概述
   2. 声明和帮助
   3. 功能模块演示
5. 总结与展望
   1. 总结
   2. 展望

参考文献

致谢

附录

第1章 绪论

* 1. 项目研究背景

现今计算机技术已经发展的很成熟了，尤其是基于windows 平台的可视化软件更是层出不穷。将图形化编程技术运用到教学中，让学生从枯燥的黑板或者幻灯片中解放出来，通过图形形象的去观察，轻松掌握知识点。

离散数学是所有计算机相关专业学生必修的一门基础课。因为离散数学和计算机息息相关。无论是硬件的存储，编码还是操作系统，编译程序等等都离不开它。通信工程中的数字信号处理，信息论与编码，无线通信等等也都会或多或少用到离散数学的知识。所以其在大学学科中的重要地位不言而喻。

但是另一方面，众所周知，这是一门逻辑思维较强、抽象难懂的学科。定义诸多，推理复杂，学生学起来吃力，老师教学的过程中也感觉有难度，对学生掌握的情况也很不满意。传统的教学中，老师播放幻灯片，学生一扫而过，之后什么印象都没有留下。这种授课方式效率不高。

把图形化编程技术和离散数学教学结合起来，实现一个可交互的动态的离散数学演示系统，让学生也可以自己动手操作，老师教授时也增强交互性。学生兴趣高，而且容易直观学习掌握离散数学的关键知识点。

* 1. 国内外研究现状

网上关于计算机图形化辅助教学工具，还是比较多的。对于计算机系而言，比较多的有数据结构演示系统，算法演示系统等等。但是关于离散数学的演示系统还是很少见的。国内有部分教师自己实现基于浏览器（javascript）的一些演示功能，缺点是不完善，而且依赖浏览器。

在离散数学的一般教学过程中，都是老师通过板书和播放幻灯片实现。大部分课件都是繁琐的定义和冗长的推理过程。学生听课时很容易昏昏欲睡。

本系统采用 C++语言开发，直接运行于windows平台，携带方便，而且提供了离散数学的动态演示功能。

* 1. 任务概述

项目的主要研究工作包括以下几个方面：

1. 根据离散数学课程和毕设任务书，对系统的总体进行设计，得到系统的总体框架。
2. 功能理论分析与实现。分析系统要实现的四个功能，每一部分用到的算法，并通过面向对象的方法，用图形化的方式展示出来。
3. 设计系统的演示界面。包括主界面，帮助和声明界面。以及具体的每一个功能的演示界面。用户可以随时查看随时使用。
   1. 论文的结构

本文的具体组织结构如下：

第1章：绪论。简单的介绍了本项目的一些相关情况。

第2章：系统概要设计。这部分主要是设计相关。介绍了开发环境和开发工具，然后进行需求分析，设计出系统的模块，以及对系统的界面进行设计。

第3章：功能模块分析与实现。这部分主要是实现相关。介绍了四个核心功能模块使用到的算法，以及代码实现的关键。

第4章：系统演示。这部分主要是和系统的演示和使用相关。

第5章：总结与展望。对本系统的设计与实现进行总结，并提出改进意见。

第2章 系统概要设计

* 1. 面向对象的编程思想
     1. 面向对象简介

面向对象编程，英文简称OOP（object oriented programing）。与传统的强调算法的过程性编程不同，它强调的是数据，其理念是设计与问题的本质特性相对应的数据格式。OOP程序设计的方法首先是设计类，也就是现实事物的一个抽象，他们准确表示了程序要处理的东西。比如一个学生类，包含了数据成员（学号，姓名，电话，成绩等）和方法（求总成绩，求平均成绩，排序等）。

面向对象，具体的来说涉及抽象、继承、多态等设计方法。如果引申的话，可以理解为一种设计思想，这种设计思想已经渗透到软件设计的各个方面，如数据库系统，交互式界面，应用结构，分布式系统，人工智能等领域。面向对象的思想又可分为面向对象分析，面向对象设计以及我们经常说的面向对象编程。

采用面向对象的设计方法，优点有很多。易维护，因为设计结构清晰，可读性高，而且由于继承的存在，即使改变需求，维护也只发生在局部模块，所以维护起来是方便且低成本的。质量高，在设计时，可重用现有的，在以前的项目领域中已被测试过的类来使系统满足业务需求并具有较高的质量。效率高，在软件开发时，根据设计的需要对现实世界的事物进行抽象来生成类，使用这样的方法解决问题，接近于日常生活和自然界的思考方式，势必提高软件开发的效率和质量。易扩展，由于继承、封装、多态的特性，自然设计出高内聚、低耦合的系统结构，使得系统更灵活、更容易扩展，而且成本较低。

* + 1. C++ 简介

和C语言一样，C++也是在贝尔实验室诞生的，Bjarne Stroustrup于20世纪80年代在这里开发出了这种语言。C++是C语言的超集，这意味着任何有效的 C 程序都是有效的 C++ 程序。它们之间有些细微差异，但无足轻重。OOP 部分赋予了 C++ 语言将问题所涉及的概念联系起来的能力，C部分则赋予了C++语言紧密联系硬件的能力。C++正是在C的语言基础上添加面向对象的特性，使得该语言更加丰富，并获得了一定程度的成功。

了解过一些编程语言，如 C、C++、java、php等的同学可能都有体会，C++是最复杂的编程语言之一。事实也的确是这样，因为它要兼C，所以不能像java那样完全面向对象。C++ 有很多知识是让人很容易困惑的，比如引用、多重继承、模板等。但是这并不影响它的杰出特性，代码结构清晰，执行效率高，多种平台支持。C++用于桌面图形化编程是最好不过的了，因为操作系统本身就是使用C和C++编写的。也无需安装其他的运行环境，java 程序就必须安装jre ，程序才可以跑的起来。

* + 1. Qt 简介

Qt是奇趣科技公司于1991年开发的一个跨平台的C++图形用户界面应用程序框架。它拥有应用程序开发者建立美观的图形用户界面所需要的所有功能。基本上，Qt同Windows平台上的 MFC，ATL 是同类型的东西。Qt有开源版本，有完整的一套开发环境，编辑器，开发文档，GUI设计师以及丰富的API。

Qt是基于C++的，采用良好的封装机制使得其模块化程度非常高，可重用性好，对于用户开发来说是非常方便的。而且跨平台，windows下使用Qt写的程序，只需在Linux下重新编译，即可实现移植。Qt开发过程中有两个概念很重要，一个是布局，另外一个是信号和槽。布局就是界面设计，水平和垂直方向的组合和对齐等关系。信号和槽则是窗口程序上响应用户操作的关键。

Qt程序的编译方式有两种静态编译和动态编译。动态编译是缺省的方式，但是这种方法编译出来的可执行文件依赖很多运行库，直接拷贝exe文件到其它电脑，程序往往会无法执行。所以这里Qt是采用的静态编译的方式安装的，因此项目编译后得到的可执行文件也是静态的（解决了所有库的依赖关系，打包到一个exe文件中），可在windows平台中任意拷贝使用。

* + 1. Git 简介

首先说明一下开发的过程中为什么要使用版本控制工具。开发的过程中可能会经常遇到这些问题：代码散落各处，整理困难；操作失误，代码无法恢复或者回滚到上一个状态很繁琐；想法改变，难以进行代码重构；多份代码并存，无法确定最新版本。使用版本控制工具，这些问题就迎刃而解了。版本控制工具又有很多，比如cvs、svn、git等。本项目开发采用Git。

Git是一款非常优秀的、开源的、免费的分布式版本控制系统，用于软件开发过程中的源码版本控制，适合应用于各种类型的项目。简单的来说就是管理源代码的各个版本，在不用版本控制工具的时候，我们可能要拷贝整个目录来做个备份，这样一个项目下来可能有好多个目录，而且每一次拷贝备份也没有更多的说明信息。使用Git就可以避免这样的问题，而且Git支持远端操作，本地的版本仓库可以同步到远端。这样即使word崩溃了或是不小心删除了文件，都可以轻松恢复。

Git不仅仅可以用来管理源代码文件，理论上所有的文件都可以。比如论文，程序都是可以的。Git的具体使用可以参考《Pro Git》一书。本项目开发过程中，使用Git做版本控制，同时同步到github.com。

* 1. 系统设计
     1. 系统的总体结构

系统总体结构如图2-1所示。打开主界面后，主窗体有菜单栏（包含关于菜单和查看帮助菜单）、工作区（包含四个操作选项）。

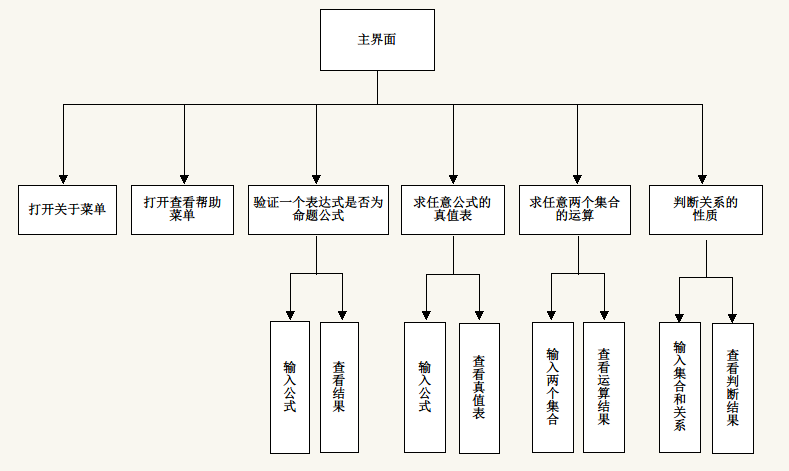


图2-1 系统总体结构图

* + 1. 系统的演示结构

本系统是采用面向对象的设计方法编写的。根据系统需求，整个系统分为两个层次，主界面和演示界面。主界面部分，有菜单栏和四个选项卡，分别对应每个要求的功能。演示界面部分，是系统的核心，用户使用时只需按照界面的提示进行输入。程序自动会把结果显示给用户。

查看帮助菜单提供了一个类似于小型浏览器的功能，可以在其中浏览、跳转、返回，针对每一块都有相应的帮助说明，在使用出现问题的时候即可打开帮助菜单，基本所有的问题都会说明。

演示界面一般大概是由三部分构成。说明部分，简单的说明该离散数学用到的知识点。输入部分，一般是一个文本框，供用户输入使用，在旁边有输入格式提示以及输入确认按钮。结果部分，用户输入完成后按下回车或者确认按钮，系统会将运行的结果以文字或表格的形式反馈给用户。演示界面支持重复使用，每次进行新的输入，无需关闭窗口并重新打开。但是每种演示界面又有所差异，所以可能不全是三部分组成。

整个离散数学演示系统架构简单、层次清晰、操作方便。用户只需点击以及配合键盘进行少量的数据输入，就可达到预期的结果。

* 1. 系统界面设计

主窗口部分，继承自QMainWindow类。窗口标题为离散数学动态演示系统。顶部的菜单栏有两个主选项卡，文件和帮助。文件选项卡下是退出子菜单，支持快捷按键Ctrl+X。帮助选项卡下有两个子菜单，关于和查看帮助。关于窗口主要是版权声明，以及作者介绍和代码地址，继承自QDialog类。查看帮助菜单窗口是一个类似于浏览器的窗口，使用QTextBrowser类，主要是针对系统的使用提供简单的帮助。主窗口的主体部分，是四个按钮，每个按钮对应一个核心功能。大概外形如图2-2所示。

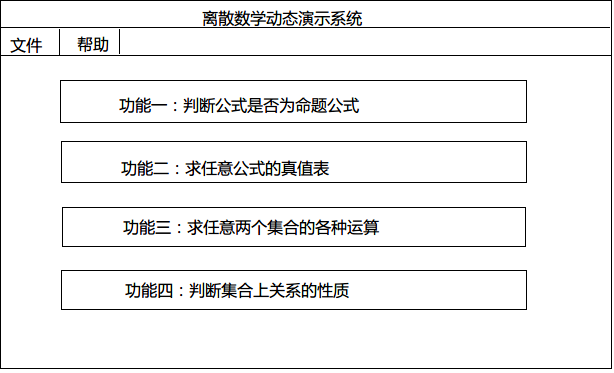


图2-2 主窗口概念图

验证一个表达式是否为命题公式窗口，继承自QDialog类。由三个面板构成，首先是说明面板，给出了命题公式的定义。然后是输入面板，有一个输入框，供用户输入公式，旁边是标注，因为命题公式定义中的某些符号，比如合取、析取、蕴含、等值，键盘上是没有这些按键的，所以做了替代。最后是结果面板，会动态的提示用户输入表达式、是命题公式、不是命题公式。大概外形如图2-3所示。

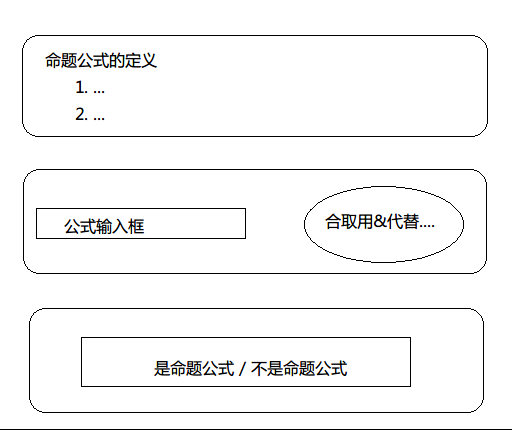


图2-3 判断公式是否为命题公式概念图

求任意公式的真值表窗口，继承自QDialog类。由两个面板构成，首先是输入面板，有一个输入框，供用户输入命题公式，旁边是标注。这个面板和验证一个表达式是否为命题公式的输入面板外形相似，但是多了输入控制功能。因为只有在用户输入的公式为合法的命题公式的时候，才能求其真值表。所以程序作了控制，如果用户输入了非法公式，会弹出警告框，提示用户重新输入正确的命题公式。然后是结果面板，命题公式的真值表是以表格的形式显示在结果面板中的，由于命题公式长度不一，所以得到的表格大小、外形会有所差异，不过并不影响结果。大概外形如图2-4所示。



图2-4 求任意公式的真值表概念图

求任意两个集合的运算窗口，继承自QDialog类。由四个面板构成，首先是说明面板，给出了集合运算的定义，主要是交集、并集、差集。然后是集合A输入面板，用户根据提示的格式要求，输入集合A。其次是集合B输入面板。最后是结果面板，将集合A和B的交集、并集、差集显示在这里。大概外形如图2-5所示。

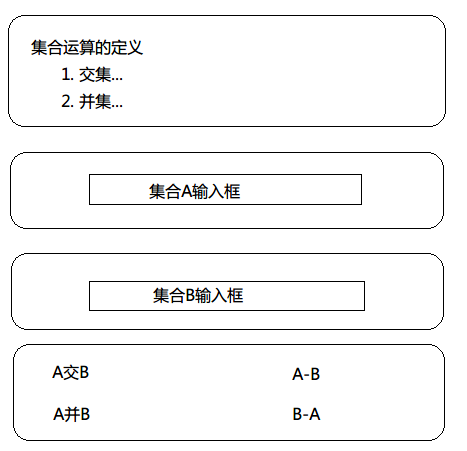


图2-5 求两个集合的各种运算概念图

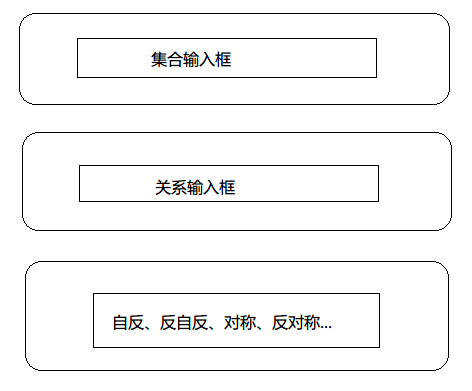
 判断关系的性质窗口，继承自QDialog类。由三个面板构成，首先是集合输入面板。然后是关系输入面板。最后是结果面板，程序会将该集合上的关系的性质显示在这里，比如自反、反自反、对称等。大概外形如图2-6所示。

图2-6 判断关系的性质概念图

界面部分主要是用到Qt提供的各种可视化组件类，如QWidget、QMainWindow、QLabel等等。可以用Qt设计师设计界面，也可以自己写代码控制界面。

* 1. 系统基本功能

1. 提供类似浏览器的帮助功能。
2. 验证一个表达式是否为命题公式。表达式是任意输入的字串。
3. 求任意公式的真值表。只要公式是合法的命题公式。
4. 求任意两个集合的运算。两个集合可以是各种形式的，数字、字母，甚至文字均可。
5. 判断关系的性质。只要输入的集合以及集合上的关系合法。

在上述各个系统功能中，充分展示了图形界面设计的特点，实现人机交互，提高了学生学习的兴趣。只要用户根据系统提示进行操作，无需进行大量的运算画图等，有程序自动给出结果，快速简便。并且系统支持扩展，以后如果要添加新的功能，只需在代码中添加新的模块既可。

* 1. 系统主要模块
     1. 主窗口

主窗口类继承自QMainWindow类，顶部的菜单栏使用QMenuBar类，子菜单使用QMenu类，主窗体中提供的四个选项使用的是QPushButton类。当用户点击事件发生的时候，响应这些事件，用到的机制是信号和槽。简单的理解就是，单击信号函数会执行，会有相应的槽函数跟着执行，响应该事件。主窗口主要是集合整个系统的主要功能。

* + 1. 声明窗口

声明窗口类继承自QDialog类，当用户点击顶部的菜单栏关于选项时弹出。面板使用QFrame类，作者和代码说明信息使用的是QLabel类。布局使用的是QVBoxLayour类，用于水平布局。声明窗口没有交互功能，用户可以根据其给出的代码地址，自行到网上下载。

* + 1. 帮助窗口

帮助窗口类继承自QDialog类，当用户点击顶部的菜单栏查看帮助选项时弹出。类似浏览器的功能是通过QTextBrowser类实现，以及上边的返回、主页按钮是通过QPushButton类实现。帮助窗口的内容可以在几个页面中跳转，分别对应不同的帮助主题，其实每一个页面的实现是一个网页文件，完全使用html页面编写。

* + 1. 功能模块

四个主要功能模块，因为每一个的逻辑都有所不同，主要是输入、输出结果格式差异很大，所以无法做到统一。每一个都继承自QDialog类，当用户单击相应的功能时弹出。后台处理时，拿到用户的数据，首先进行合法性检查，然后用所对应的算法实现，得到结果后输出在窗口界面。

第3章 功能模块分析与实现

* 1. 验证表达式是否为命题公式

要理解离散数学中的命题公式的定义，先要掌握离散数学中的五种基本连接词（也称逻辑运算符），其分别是合取（）、析取（）、蕴含（）、等值（）、否定（┐）。这五个基本连接词是命题公式的基础元素之一，并且具有不同的优先级，在计算命题公式的值的时候，优先级会起到很大的影响作用。离散数学中命题公式的定义为：

1. 单个原子公式是命题公式
2. 如果A和B是命题公式，则（┐A），（A），（A），（A），（A）是命题公式。
3. 只有有限步的应用规则（1）和（2）生成的公式，才是命题公式。

上述这种命题公式的定义为归纳定义，也可以叫做递归定义。我们一般直接判断给定的一个公式是否为命题公式时，就是对字串应用以上规则，如果符合，则为命题公式，否则就不是命题公式。但是在程序中如果也使用这种方法会非常复杂，所以采用了生成公式树的方法判定。

先举几个简单的例子，看看如何判断一个公式是否为命题公式。如（P （P ⋁Q）），根据规则（1），P是命题公式，Q是命题公式。根据规则（2），（P⋁Q）是命题公式，再由规则（2），得到（P （P ⋁Q））是命题公式。再看一个反面的例子，PQ⋀R，不能由上述三条形成规则得出，所以不是命题公式。

程序处理的过程如下（输入时合取用 & 代替，析取用 | 代替，蕴含用 - 代替，等值用= 代替，否定用 ！代替）：

1. 获取用户输入的字符串。因为用户输入的字串具有任意性，所以先进行正则匹配（正则表达式是对字符串操作的一种逻辑公式，也即利用预先定义好的一些特定字符、及这些特定字符的组合，形成一个“规则字符串”，然后使用这个“规则字符串”去对目标字符串进行过滤），如果字串只含有a-z、A-Z、&、|、！、（、）、-、=、空白，则是合法的字串，否则直接给出结果：不是命题公式。这个功能的实现可直接使用Qt提供的QRegExp类，该类中有关于正则匹配的方法。
2. 去除字符串中的空格，便于后续处理。
3. 构建一颗公式树，树的结点有左右两个孩子。树的父节点为公式中的连接词，树的叶子结点为命题变元a-z、A-Z。如果可以构建这样的一棵树，那么该公式为命题公式，否则就不是命题公式。构建的过程中涉及连接词的优先级问题，单独调用函数判断。
4. 将结果返回给界面，界面程序会在结果面板给出是否为命题公式的判定。

整个算法的核心在于构建这样一颗公式树，叶子节点都为命题变元，非叶子节点都是命题连接词。公式树构建的过程即是判断公式是否为命题公式的过程。实现的源码文件在module目录下的moduleonefunction.cpp中。

* 1. 求任意公式的真值表

命题公式的真假值一般是不确定的。给所有的命题变元赋值，在所有的赋值组合下，得到命题公式取值情况所列成的表称之为真值表。如课本上给出的一个例子：

表3-1 （（P⋁Q）⋀ P）的真值表

|  |  |  |
| --- | --- | --- |
| P | Q | （（P⋁Q）⋀ P） |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

对于含有n（n≥1）个命题变元的命题公式，共有 组赋值。程序的关键就是求得在每一组命题变元的赋值下，命题公式是真还是假（1或者0）。然后将得到的每一行数据组合在一起就生成了命题公式的真值表。

程序处理的过程如下（输入时合取用 & 代替，析取用 | 代替，蕴含用 - 代替，等值用= 代替，否定用 ！代替）：

1. 获取用户输入的字符串，直接调用上一个模块，判断公式是否为命题公式的方法，判断输入的字符串的合法性。
2. 将命题公式中的命题变元保存在容器vector<char> varNames中，同时调用方法将表达式转换成后缀形式保存起来。所谓后缀表达式就是：不包含括号，运算符（连接词）放在两个运算对象（命题变元）的后面，所有的计算按运算符出现的顺序，严格从左向右进行（不再考虑运算符的优先规则）。转化的过程如下：对命题公式开始扫描；遇到命题变元，加入后缀表达式；遇到连接词，若为左括号，入栈。若为右括号，则依次把栈中的连接词加入到后缀表达式中，知道出现左括号为止，同时删除栈中的左括号；若是非括号外的其他连接词，当其优先级高于除左括号以外的栈顶连接词时，直接入栈。否则从栈顶开始，依次弹出比当前处理的连接词优先级高和优先级相等的连接词，直到一个比它优先级低的或者遇到了一个左括号为止。当扫描的命题表达式结束时，栈中的的所有连接词出栈。举个例子，比如 P∨Q∧R，转换成后缀表达式就是PQR∧∨。
3. 取一组命题变元的赋值组合，对后缀表达式进行计算。计算过程如下：从左到右遍历后缀表达式的每个数字和连接词，遇到是命题变元就进栈，遇到是连接词，就将处于栈顶的两个命题变元出栈，进行运算，运算结果进栈，一直到最终获得结果。
4. 使用循环，遍历所有的命题变元赋值组合，得到命题公式的结果。保存在结果容器中。
5. 显示界面将变元，命题公式，赋值组合，以及命题公式的值以表格的形式显示在结果面板中。

整个算法的核心在于中缀表达式（原命题公式）到后缀表达式的转换以及对后缀表达式的计算。其中要用到数据结构中的栈结构做辅助，得到的结果由于是一行行的数据，所以保存在Qt提供的容器类模板vector中。实现的源码文件在module目录下的moduletwofunction.cpp中。

* 1. 求任意两个集合的运算

离散数学中对集合的定义为：一个集合是能作为整体论述的事物的集体。组成集合的每个事物叫做这个集合的元素或成员。通常用大写字母A，B，C代表集合；用小写字母a，b，c代表元素。集合的运算定义如下：

设A和B是集合，

1. A和B的并集记为AB，是集合。

AB={x|xA或xB}

1. A和B的交集记为AB，是集合。

AB={x|xA且xB}

1. A和B的差，或B关于A的相对补，记为A-B，是集合。

A-B={x| x|xA且xB }

程序的处理过程如下：

1. 获取用户输入的集合（本质上是字符串），将集合中的元素提取出来分别保存在两个集合对象中。
2. 对两个集合对象分别使用不同的芳法，求其交集、并集、补集。运算的结果保存在几个新的集合对象中。
3. 显示界面将集合对象转换成集合的形式，显示在结果面板的相应位置处。

求两个集合的交集运算的算法流程为：建立一个哈希表（HashTable），其键（KEY）表示集合中数字的值，其值（VALUE）表示集合中数字出现的次数。遍历集合A，将集合中的每个数字（KEY）插入哈希表，每个数字的出现次数（VALUE）设置为1。遍历集合B，对于集合中的每个数字：如果哈希表中已经存在该数字，将对应的VALUE改为2；如果哈希表中不存在该数字，忽略。遍历哈希表，输出VALUE为2的数字，即得到A和B的交集。

求两个集合的并集运算的算法流程为：建立一个哈希表（HashTable），其键（KEY）表示集合中数字的值，其值（VALUE）可以忽略。.遍历集合A，将集合A中的每个数字（KEY）插入哈希表。遍历集合B，对于集合中的每个数字：如果哈希表中已经存在该数字，忽略。如果哈希表中不存在该数字，将这个数字插入哈希表。遍历哈希表，输出哈希表中的每个KEY，即得到A和B的并集。

求两个集合的差集运算的算法流程为：建立一个哈希表（HashTable），其键（KEY）表示集合中数字的值，其值（VALUE）表示集合中数字出现的次数。遍历集合A，将集合中的每个数字（KEY）插入哈希表，每个数字的出现次数（VALUE）设置为1。遍历集合B，对于集合中的每个数字：如果哈希表中已经存在该数字，将对应的VALUE改为2；如果哈希表中不存在该数字，忽略。遍历哈希表，输出VALUE为1的数字，即得到A和B的差集。

Qt作为一款优秀的类型库，提供了大量的“容器类”，专门用于以某种方式存储大量内容。其中就包括集合类QSet，它的实现就是使用哈希表。在程序中只需要定义两个QSet对象用来保存用户输入的两个集合，然后对这两个对象进行+、&、- （操作符重载过）操作，分别对应并集、交集、差集，即可得到一个新的集合对象，也就是我们要的结果。实现的源码文件在module目录下的modulethreefunction.cpp中。

* 1. 判断关系的性质

离散数学中对二元关系的定义为：A×B的子集叫做A到B的一个二元关系。表达有限集合到有限集合的二元关系的时候，关系矩阵是一有力的工具。集合上关系的性质定义如下：

设R是集合A上的二元关系，

1. 如果对A中每一x，xRx，那么R是自反的。即

A上的关系R是自反的∀x（x∈A→xRx）

1. 如果对A中每一x，x~~R~~x，那么R是自反的。即

A上的关系R是自反的∀x（x∈A→x~~R~~x）

1. 如果对每一x，y∈A，xRy蕴含着yRx，那么R是对称的。即

A上的关系R是对称的⟺∀x∀y（x∈A ∧ y∈A ∧ xRy→yRx）

1. 如果对每一x，y∈A，xRy，yRx蕴含着x=y，那么R是反对称的。即

A上的关系R是反对称的⟺∀x∀y（x∈A ∧ y∈A ∧ xRy ∧ yRx→x=y）

1. 如果对每一x，y，z∈A，xRy，yRz蕴含着xRz，那么R是传递的。即

A上的关系R是传递的⟺∀x∀y∀z（x∈A ∧ y∈A ∧ z∈A ∧ xRy ∧ yRz→xRz ）

关系的性质可能是以上五种中的几种性质的组合。程序中无法从定义直接去判断性质，所以借用了二元关系矩阵的特点去判断关系具有哪些性质。

程序的处理过程如下：

1. 获取用户输入的集合和集合上的关系。
2. 根据集合和集合上的关系生成关系矩阵存储在二维数组中。
3. 依次判断关系矩阵（二维数组）的特点，得到关系的性质。
4. 显示界面将步骤3得到的性质字符串拼接在一起，然后输出在结果面板中。

关系具有自反性时，关系矩阵的特点：如果该关系对应的邻接矩阵的主对角线元素都为1，则该关系具有自反性。

关系具有反自反性时，关系矩阵的特点：如果该关系对应的邻接矩阵的主对角线元素都为0，则该关系具有反自反性。

关系具有对称性时，关系矩阵的特点：如果该关系对应的邻接矩阵具有对称性，即邻接矩阵关于主对角线对称，则该关系具有对称性。

关系具有反对称性时，关系矩阵的特点：如果该关系对应的邻接矩阵除主对角线元素之外的任何元素R[i][j]=1时，R[j][i]必等于0，则该关系具有反对称性。

关系具有传递性时，关系矩阵的特点：如果该关系对应的邻接矩阵当元素R[i][j]=1且R[j][k]=1时，必有R[i][k]=1，则该关系具有传递性。

整个算法的核心在于，根据用户输入的集合和集合上的二元关系去构造一个关系矩阵（邻接矩阵），然后根据关系矩阵的特点判断关系所具有的性质。实现的源码文件在module目录下的modulefourfunction.cpp中。

第4章 系统演示

* 1. 系统概述
  2. 声明和帮助
  3. 功能模块演示

第5章 总结与展望

* 1. 总结
  2. 展望