

**实 验 报 告**

**（2019 / 2020 学年 第 一 学期）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 离散数学 | | | | | |
| 实验名称 | 利用真值表法求取主析取范式以及主合取范式的实现 | | | | | |
| 实验时间 | 2019 | 年 | 10 | 月 | 1 | 日 |
| 指导单位 | 计算机学院计算机科学与技术系 | | | | | |
| 指导教师 | 陈兴国 | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 | 吴雯 | 班级学号 | B18030322 |
| 学院(系) | 计算机学院 | 专 业 | 计算机科学与技术系 |

**实 验 报 告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | 利用真值表法求取主析取范式以及主合取范式的实现 | | | **指导教师** | 陈兴国 |
| **实验类型** | **验证** | **实验学时** | **4** | **实验时间** | 2019.10.1 |
| 1. **实验目的和要求**   通过编程实现主析取范式以及主合取范式的真值表求法以巩固相关理论的掌握  要求：  1).输入合式公式  2).规范列出所输入的公式的真值表  3).给出相应主析取和主合取范式 | | | | | |
| 二、**实验环境(实验设备)**  硬件：微型计算机  软件：Windows 操作系统、Microsoft Visual C++6.0、Java等可视化编程语言 | | | | | |
| **三、实验原理及内容**  1、内容:编程实现用真值表法求任意含三个以内变量的合式公式的主析取范式和主合取范式。  原理:首先读入变元个数，然后读入合式公式，用堆机的知识将中缀表达式转化为后缀表达式，调用查定、析取、合取、条件、双条件的函数计算元素取不同真值时合式公式的真值，然后输出真值表，调用计算主析取范式和主合取范式的函数并输出。  2、给出核心算法的C++或Java等语言的源代码，并加上详细注释，分析算法的时间复杂度；  //以下为纯vs中c++代码，Qt应用代码未贴出  // 或运算 | , 与运算 & ,单条件 -> ,双条件 <=> ,非运算 !  #include <cstdio>  #include <iostream>  #include <vector>  #include <string>  #include <cstdlib>  #include <queue>  #include <stack>  #include <map>  #include <sstream>  using namespace std;  string zhong; //中缀表达式  char hou[1000]; //后缀表达式  string alpha; //存放所有字母变量  map<char, int> M; //映射，将字母变量与0或1一一对应  struct note  {  int a[100];  };  vector<note> True; //不定长数组，存放主析取范式对应字母变量的01情况，也就是表达式真值为T  vector<note> False; //不定长数组，存放主合取范式对应字母变量的01情况，也就是表达式真值是F  void ddd() //预处理，去除中缀表达式中条件->中的>,和双条件<=>中的= and > ,将这两个运算符当成一个字符处理，更方便  {  string::iterator i = zhong.begin();  int flag = 1;  while (flag)  {  flag = 0;  for (i = zhong.begin(); i != zhong.end(); ++i)  {  if (\*i == '>')  {  zhong.erase(i);  flag = 1;  break;  }  if (\*i == '=')  {  zhong.erase(i);  flag = 1;  break;  }  if (\*i == '!'&&\*(i + 1) == '!')  {  zhong.erase(i);  zhong.erase(i);  flag = 1;  break;  }  }  }  }  int icp(char a) //栈外优先级  {  if (a == '#') return 0;  if (a == '(') return 12;  if (a == '!') return 10;  if (a == '&') return 8;  if (a == '|') return 6;  if (a == '-') return 4;  if (a == '<') return 2;  if (a == ')') return 1;  }  int isp(char a) //栈内优先级  {  if (a == '#') return 0;  if (a == '(') return 1;  if (a == '!') return 11;  if (a == '&') return 9;  if (a == '|') return 7;  if (a == '-') return 5;  if (a == '<') return 3;  if (a == ')') return 12;  }  void change() //中缀表达式转换后缀表达式  {  int j = 0;  stack<char> s;  char ch, y;  s.push('#'); //添加#  //char t1, t2;  stringstream ss(zhong);  while (ss >> ch, ch != '#') //string转char  {  if (isalpha(ch)) //如果参数是字母返回非0，否则返回0  {  hou[j++] = ch;  if (alpha.find(ch) == -1)  {  alpha.push\_back(ch);  }  }  else if (ch == ')')  {  for (y = s.top(), s.pop(); y != '('; y = s.top(), s.pop()) //pop()返回void，top()返回栈顶引用  {  hou[j++] = y;  }  }  else  {  for (y = s.top(), s.pop(); icp(ch) <= isp(y); y = s.top(), s.pop())  {  hou[j++] = y;  }  s.push(y);  s.push(ch);  }  }  while (!s.empty()) //判断栈空  {  y = s.top(); //访问栈顶  s.pop(); //出栈，删除栈顶元素，不返回该元素  if (y != '#')  {  hou[j++] = y;  }  }  hou[j] = '#';  }  int cal() //对赋值后的后缀表达式进行计算  {  stack<int> s;  char ch;  int j = 0;  int t1, t2;  while (1)  {  ch = hou[j];  if (ch == '#') break;  if (ch == 0) break;  j++;  if ((ch >= 'A'&&ch <= 'Z') || (ch >= 'a'&&ch <= 'z'))  {  s.push(M[ch]); //入栈，栈顶添加元素  }  else  {  if (ch == '!')  {  t1 = s.top();  s.pop();  s.push(!t1);  }  else if (ch == '&')  {  t1 = s.top();  s.pop();  t2 = s.top();  s.pop();  if (t1 == 1 && t2 == 1)  {  s.push(1);  }  else  {  s.push(0);  }  }  else if (ch == '|')  {  t1 = s.top();  s.pop();  t2 = s.top();  s.pop();  if (t1 == 0 && t2 == 0)  {  s.push(0);  }  else  {  s.push(1);  }  }  else if (ch == '-')  {  t1 = s.top();  s.pop();  t2 = s.top();  s.pop();  if (t1 == 0 && t2 == 1)  {  s.push(0);  }  else  {  s.push(1);  }  }  else if (ch == '<')  {  t1 = s.top();  s.pop();  t2 = s.top();  s.pop();  if ((t1 == 1 && t2 == 1) || (t1 == 0 && t2 == 0))  {  s.push(1);  }  else  {  s.push(0);  }  }  }  }  int ans = s.top();  return ans;  }  void dfs(int cur) //递归枚举每一种字符变量的取值情况  {  if (cur == alpha.size())  {  int ans = cal();  for (int i = 0; i < alpha.size(); i++)  {  if (M[alpha[i]])  {  printf\_s("T\t");  }  else  {  printf("F\t");  }  }  if (ans == 1) //真值为T 计入到True数组，以待后面主析取范式使用  {  printf\_s("T\n");  note t;  for (int i = 0; i < alpha.size(); i++)  {  t.a[i] = M[alpha[i]];  }  True.push\_back(t);  }  else //真值为F 计入到False数组，以待后面主合取范式使用  {  printf\_s("F\n");  note t;  for (int i = 0; i < alpha.size(); i++)  {  t.a[i] = M[alpha[i]];  }  False.push\_back(t); //插入字符  }  return;  }  M[alpha[cur]] = 1;  dfs(cur + 1);  M[alpha[cur]] = 0;  dfs(cur + 1);  }  int main()  {  int flag = 1;  while (flag)  {  int i;  M.clear();  alpha.clear();  True.clear();  False.clear();  printf\_s("请输入表达式,回车结束\n");  cin >> zhong;  zhong.append("#");  ddd(); //符号处理  change(); //中转后缀  for (i = 0; i < alpha.size(); i++)  {  printf\_s("%c\t", alpha[i]);  }  printf\_s("表达式真值\n");  dfs(0);  printf\_s("主析取范式为\n");  for (i = 0; i < True.size(); i++)  {  if (i != 0) printf\_s("∨");  int \*p = True[i].a;  printf\_s("(");  for (int j = 0; j < alpha.size(); j++)  {  if (j != 0) printf\_s("∧");  if (p[j] == 1)  {  printf\_s("%c", alpha[j]);  }  else  {  printf\_s("￢%c", alpha[j]);  }  }  printf\_s(")");  }  printf\_s("\n");  printf\_s("主合取范式为\n");  for (i = 0; i < False.size(); i++)  {  if (i != 0) printf\_s("∧");  int \*p = False[i].a;  printf\_s("(");  for (int j = 0; j < alpha.size(); j++)  {  if (j != 0) printf\_s("∨");  if (p[j] == 0)  {  printf\_s("%c", alpha[j]);  }  else  {  printf\_s("￢%c", alpha[j]);  }  }  printf\_s(")");  }  printf\_s("\n\n");  cout << "是否需要继续，继续(1),退出(0)" << endl;  scanf\_s("%d", &flag);  if (flag == 0)  break;  else  system("cls");  }  return 0;  }  3、给出测试数据及运行结果、实验相关结论等。  测试数据：((!!!P|Q)<=>(A&B))->(R|(!B&C))  运行结果：  总界面：    真值表： | | | | | |

**实 验 报 告**

|  |
| --- |
| 主合取范式：    主析取范式： |

**实 验 报 告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **四、实验小结**（包括问题和解决方法、心得体会、意见与建议等）  **说明：这部分内容主要包括：在编程、调试或测试过程中遇到的问题及解决方法、本次实验的心得体会、进一步改进的设想等。**  (一)实验中遇到的主要问题及解决方法  刚开始思考如何写这个程序的时候，我不知道该如何处理输入进来的公式，在翻阅了数据结构书籍之后我才忽然反应过来，可以用堆栈的知识，将输入进来的中缀表达式转化为后缀表达式，这样就方便计算合式的真值了。  （二）实验心得  这次离散数学实验中，我成功地将所学习得C++的编程知识和数据结构里的堆栈的知识运用了起来，虽然中途遇到了一些困难，但最终都很好地解决了。这次实验让我学会了利用编程语言来求主析取范式和主合取范式，更加深刻地理解了这两种范式，并且也让我对C、C++等编程语言有了更强的运用能力，让我明白了离散数学和编程知识是息息相关、密不可分的。以后我将更加认真学习离散数学，并且更多地将编程的知识运用起来。  （三）意见与建议（没有可省略） | | | | | |
| 1. **支撑毕业要求指标点**   支撑毕业要求的指标点为：   * 1-4掌握计算机科学与技术领域的专业知识，能将专业知识用于分析和解决计算机领域复杂工程问题。   √   * 2-1能够应用数学、自然科学和工程科学的基本知识，识别和分析计算机领域复杂工程问题的特征。 | | | | | |
| **六、指导教师评语 (含学生能力达成度的评价)** | | | | | |
| **成 绩** |  | **批阅人** |  | **日 期** |  |

如果不太想写太多字，“指导教师评语”也可以设计为如下的各选择项用打勾形式（仅仅作为一个简单示例，请各课程负责人根据课程和实验情况以及支撑的指标点来自行设定选择项，同一门课程的不同实验评分细则项允许存在不同）：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **评 分 细 则** | **评分项** | **优秀** | **良好** | **中等** | **合格** | **不合格** |
| **遵守实验室规章制度** |  |  |  |  |  |
| **学习态度** |  |  |  |  |  |
| **算法思想准备情况** |  |  |  |  |  |
| **程序设计能力** |  |  |  |  |  |
| **解决问题能力** |  |  |  |  |  |
| **课题功能实现情况** |  |  |  |  |  |
| **算法设计合理性** |  |  |  |  |  |
| **算法效能评价** |  |  |  |  |  |
| **回答问题准确度** |  |  |  |  |  |
| **报告书写认真程度** |  |  |  |  |  |
| **内容详实程度** |  |  |  |  |  |
| **文字表达熟练程度** |  |  |  |  |  |
| **其它评价意见** |  | | | | |
| **本次实验能力达成评价（总成绩）** |  |  |  |  |  |