

**实验报告**

**（2019 / 2020学年第一学期）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 离散数学 | | | | | |
| 实验名称 | 盖住关系的求取及格的判定 | | | | | |
| 实验时间 | 2020 | 年 | 12 | 月 | 2 | 日 |
| 指导单位 | 计算机学院计算机科学与技术系 | | | | | |
| 指导教师 |  | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 |  | 班级学号 |  |
| 学院(系) | 计算机学院 | 专业 | 信息安全 |

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | 盖住关系的求取及格的判定 | | | **指导教师** | 柯昌博 |
| **实验类型** | **验证** | **实验学时** | **4** | **实验时间** | 2020.12.2 |
| 1. **实验目的和要求**   目的：通过编程实现偏序关系中盖住关系的求取及格论中有补格的判定以巩固相关理论的掌握  要求：从屏幕输入一个正整数，列出由该整数所有因子构成的盖住关系，给出由其因子构成的格是否为有补格的判定。 | | | | | |
| 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、C++ | | | | | |
| **三、实验原理及内容**  **主要数据结构和算法**  辗转相除法求最大公约数  Lcm （i,j）= [i]\*[j] / Gcd(i,j)求最小公倍数  **原理**  1.Element()函数通过遍历的方式找到正整数n的所有k个因子。  2.设置一个k\*k的二维关系数组，将这k个因子之间存在的整除关系录入数组，如果满足i | j，则matrixs[i][j] = 1。  3.显然这k个因子是一个偏序集，通过cover（）函数求其盖住关系。  4.因为盖住关系存在最小上界和最大下界，因此是一个格。  5.判断是否是有补格。对于任意一个因子，遍历其他因子，求他们的最小上界和最大下界（最小公倍数和最大公约数）是否是最大元（n）和最小元（1）。若是，则为有补格。  **完整代码+注释**  #include <iostream>  #include <cstring>  #include <cstdio>  #include <string>  using namespace std;  int count = 0; //从 0 开始计数  int n; //正整数  int elements[100]; //存因子  int matrixs[100][100] = {0}; //初始化为 0  void element()//分解n的因数  {  cout << "整数 " << n << " 的因子: ";  for(int i = 1; i <= n/2; i++){  if(n % i == 0){  elements[count++] = i;  cout << i << ", ";  }  }  elements[count] = n;  cout << n << endl << endl;  }  //盖住关系  void cover(){  //关系矩阵初始化  for(int i = 0; i < count+1; ++i){  for(int j = 0; j < count+1; ++j){  if(elements[j] % elements[i] == 0){ //如果满足整除关系，就设为 1  matrixs[i][j] = 1;  }  }  }  //开始判断  for(int i = 0; i < count+1; ++i){  for(int j = 0; j < count+1; ++j){  for(int k = 0; k < count+1; ++k){  matrixs[k][k] = 0; //先去掉自反性  if(matrixs[i][j] && matrixs[j][k]){  matrixs[i][k] = 0; //再去掉传递性  }  }  }  }  cout << "盖住关系是: {";  for(int i = 0; i < count+1; ++i){  for(int j = 0; j < count+1; ++j){  if(matrixs[i][j]){ //除去前面去掉的，其他就满足盖住关系了  cout << " <" << elements[i] << "," << elements[j] << ">";  }  }  }  cout << " }" << endl;  }  //求最大公约数  int gcd(int x, int y){  int m;  //辗转相除法  while(m != 0){  m = x % y;  x = y;  y = m;  }  return x;  }  //判断有补格  void IFcomplemented\_lattice(){  bool flag;  int Gcd, Lcm;  for(int i = 1; i < count-1; i++){ //判断有补格时不需要考虑最大元和最小元  flag = false; // 初始化 flag = false  for(int j = 1; j < count-1; j++){  if(i == j)  continue;  Gcd = gcd(elements[i], elements[j]); // 最大公约数，即最大下界  Lcm = elements[i]\*elements[j] / Gcd; // 最小公倍数，即最小上界  if(Gcd == 1 && Lcm == n){ // 若是补元，flag = true  flag = true;  break;  }  if(!flag){ // 若存在元素没有补元，则此格不是有补格  cout << "这不是有补格" << endl;  return;  }  }  }  // 若所有元素均有补元，则此格是有补格  cout << "这是有补格" << endl;  return;  }  int main(){  cout << "请输入一个正整数: ";  cin >> n;  cout << endl;  element();  cover();  cout << endl;  IFcomplemented\_lattice();  cout << endl;  return 0;  }  **运行截图** | | | | | |

**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）   1. 实验中遇到的主要问题及解决方法   逐个遍历求最大公约数和最小公倍数太过于浪费时间。我在网上查询了数论有关知识，通过辗转相除法和公式求最大公约数和最小公倍数，提高了程序效率。  （二）实验心得  1.关于有补格的判定，最大元和最小元互为补元。遍历时从下标1到count-2.  2.k个因子的整除关系必然是格，可能为有补格  3.有限格必定存在最大元和最小元。  4.时间复杂度：element（）复杂度为O（n）  Cover（）复杂度为 O（n^3）  辗转相除法求最大公约数复杂度为O（log（max（a，b）））  IFcomplemented\_lattice（）复杂度为 O（n^2 log（max（a，b）））  总时间复杂度为O（n^3）  （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）** |  |  |  |  |  |