# 第4次编程练习报告

姓名：申宗尚 学号：2213924 班级：信安班

##### 编程练习1——求解最小原根并基于最小原根构造指数表

* **源码部分：**
* #include <iostream>  
  #include <vector>  
  #include <iomanip>  
  using namespace std;  
  int m;  
  void binary(vector<int>& a, int T)  
  {  
   int q = T;  
   int r;  
   while (q != 0)  
   {  
   r = q % 2;  
   a.push\_back(r);  
   q = q / 2;  
   }  
  }  
  int pfc(int a, int n, int x)  
  {  
   vector<int> b;  
   binary(b, n);  
   int c = 1;  
   for (int i = b.size() - 1; i >= 0; i--)  
   {  
   c = c \* c % x;  
   if (b[i])  
   {  
   c = c \* a % x;  
   }  
   }  
   return c;  
  }  
  bool isprime(int n)  
  {  
   if (n == 1)  
   return false;  
   else  
   {  
   for (int i = 2; i < n; i++)  
   {  
   if (n % i == 0)  
   return false;  
   }  
   return true;  
   }  
  }  
  void fj(int n, vector<int>& res)  
  {  
   int l = 0;  
   for (int i = 2; i <= n; i++)  
   {  
   if (isprime(i))  
   {  
   if (n % i == 0)  
   {  
   l++;  
   res[i]++;  
   int t = n / i;  
   m = t;  
   fj(t, res);  
   break;  
   }  
   }  
   }  
   if (!l)  
   res[m]++;  
  }  
  int oula(int n)  
  {  
   vector<int> vec(10000, 0);  
   fj(n, vec);  
   int fai = n;  
   for (int i = 2; i < vec.size(); i++)  
   {  
   if (vec[i] != 0)  
   fai \*= (1 - 1 / double(i));  
   }  
   return fai;  
  }  
  int main()  
  {  
   cout << "Please inputn(n>0): ";  
   int n;  
   cin >> n;  
   int fai = oula(n);  
   int g = 0;  
   vector<int> exp;  
   vector<int> res(10000);  
   fj(fai, res);  
   for (int i = 2; i < res.size(); i++)  
   {  
   if (res[i] != 0)  
   exp.push\_back(fai / i);  
   }  
   for (int i = 2; i < n; i++)  
   {  
   bool flag = 1;  
   for (int j = 0; j < exp.size(); j++)  
   {  
   int t = pfc(i, exp[j], n);  
   if (t == 1)  
   {  
   flag = 0;  
   break;  
   }  
   }  
   if (flag)  
   {  
   g = i;  
   break;  
   }  
   }  
   cout << "The min primitive root of " << n << ": g=" << g << endl;  
   cout << "The ind\_table of " << n << " based on g=" << g << " is:" << endl;  
   cout << setw(6) << " ";  
   for (int i = 0; i < 10; i++)  
   cout << setw(6) << i;  
   cout << endl;  
   int row = n / 10;  
   int\*\* table = new int\* [row + 1];  
   for (int i = 0; i < row + 1; i++)  
   {  
   table[i] = new int[11];  
   table[i][0] = i;  
   for (int j = 1; j < 11; j++)  
   table[i][j] = -1;  
   }  
   for (int i = 0; i <= fai - 1; i++)  
   {  
   int t = pfc(g, i, n);  
   int row\_num = t / 10;  
   int col\_num = t % 10;  
   table[row\_num][col\_num + 1] = i;  
   }  
   for (int i = 0; i < row + 1; i++)  
   {  
   for (int j = 0; j < 11; j++)  
   {  
   if (table[i][j] != -1)  
   cout << setw(6) << table[i][j];  
   else  
   cout << setw(6) << "-";  
   }  
   cout << endl;  
   }  
   return 0;  
  }
* **说明部分：**

**这段代码实现了求一个正整数的最小原根以及它的指数表。**

**binary函数实现将一个十进制数转化为二进制数， pfc函数实现快速幂算法，isprime函数判断一个数是否是素数，fj函数实现分解质因数功能，将一个数分解成若干个素数的积的形式，并将这些素数的指数存储在 vector<int> res 中。oula函数计算欧拉函数值，**

**在main函数中，首先输入一个正整数n，然后计算fai(n)，并将fai(n)分解成若干个不同的因子，存储在vector<int> exp中。这里将指数表中的行数设为n/10，每行有10个元素，因为指数表中的元素是从0到n-1的所有非重复元素，而0可以作为第一行的元素，所以总行数为n/10+1。然后依次枚举2到n-1的整数i$，对于每个i，判断是否是n的一个原根，即对于fai(n)中的每个因子d。如果是原根，则将其存储在变量g中，并退出循环。最后使用动态数组int \*\*table存储指数表，并输出到屏幕上。动态数组的行数为n/10+1，列数为11，其中第一列存储行数，第二列到第十一列存储该行的元素。如果某个元素不存在，则用-表示。**

* **运行示例：**



