# 第 4 次编程练习报告

姓名：曹瑜　　学号：2212794　　班级：密码科学与技术

## 一、编程练习 1——求解最小原根并构造指数表

- **源码部分：**

```
#include <iostream>
#include<vector>
#include<iomanip>
using namespace std;

// 求解欧拉函数
int oula(int n) {
    int result = n;
    for (int i = 2; i * i <= n; ++i) {
        if (n % i == 0) {
            while (n % i == 0) {
                n /= i;     }
            result -= result / i;
        }
    }
    if (n > 1) {
        result -= result / n; }
    return result;
}

//判断是否为原根
bool isPrimitiveRoot(int a, int n) {
    int p = oula(n);
    int result = 1;
    for (int j = 0; j < p-1; ++j) {
        result = (result * a) % n;
        if (result == 1) {
            return false;
        }
    }
    return true;
}
```

```cpp
// 从 2 开始找最小原根
int findPrimitiveRoot(int n) {
    for (int a = 2; a < n; ++a) {
        if (isPrimitiveRoot(a, n)) {
            return a;
        }
    }
    return -1;
}

// 构造指数表
vector<vector<int>> constructExponentTable(int n, int primitiveRoot) {
    vector<vector<int>> exponentTable(n / 10 + 1, vector<int>(10, -1));
    for (int p = 0; p < n; p++) {
        int result = 1;
        for (int i = 0; i < p; i++) {
            result = (result * primitiveRoot) % n; // 使用模运算防止结果溢出
        }
        int row = result / 10;
        int col = result % 10;
        if (exponentTable[row][col] == -1) {
            exponentTable[row][col] = p;
        }
    }
    return exponentTable;
}

// 输出指数表
void printExponentTable(int n, const vector<vector<int>>& exponentTable) {
    cout << setw(8) << " ";
    for (int i = 0; i < 10; i++) {
        cout << setw(8) << i;
    }
    cout << endl;
    for (int i = 0; i <= n / 10; i++) {
        cout << setw(8) << i;
        for (int j = 0; j < 10; j++) {
            if (exponentTable[i][j] == -1) {
                cout << setw(8) << "-";
            }
            else {
                cout << setw(8) << exponentTable[i][j];
            }
        }
```

```
        cout << endl;
    }
}


int main()
{
    int n;
    cout << "Please input n(n>0): ";
    cin>>n;

    int g= findPrimitiveRoot(n);
    cout << "The main primitive root of 103: g="<<g<<endl;
    cout << "The ind_table of 103 based on g=" << g << " is:" << endl;
    vector<vector<int>> exponentTable = constructExponentTable(n, g);
    printExponentTable(n, exponentTable);

}
```

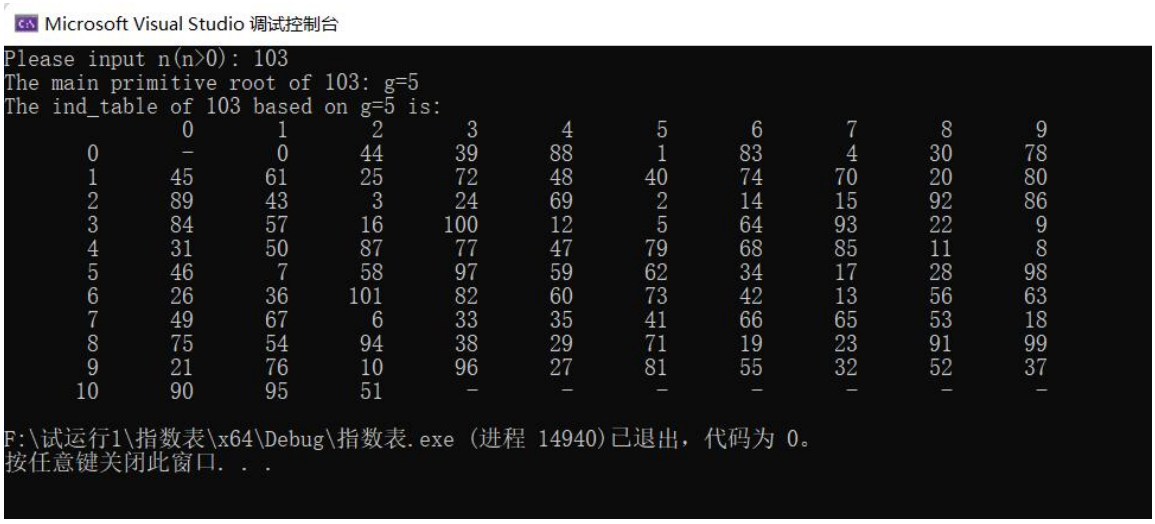
- **说明部分：**//主要说明实现的一些基本原理等

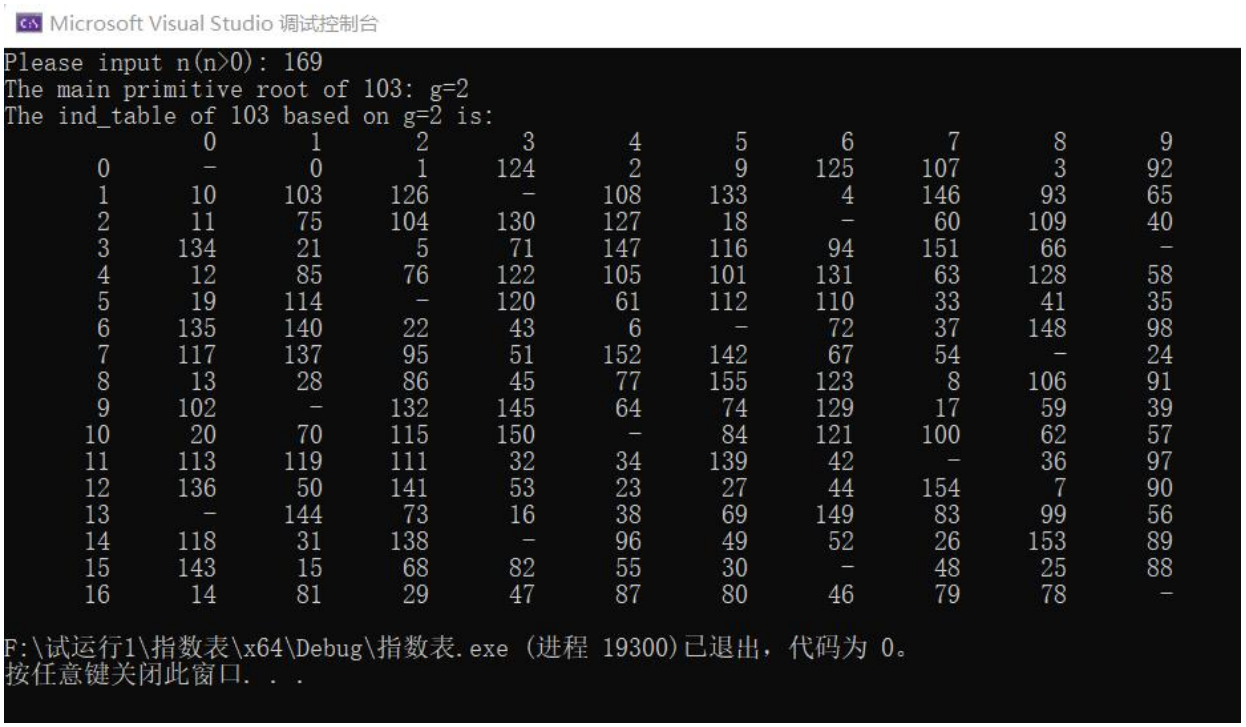
求解原根部分：从 g=2 开始依次判断是否为原根，原根判断函数中，先求解 n 的欧拉函数 p，然后验证 g 的 0-p-1 次幂模 n 是否为 1，根据最小原根定义，值为 1 则一定不是最小原根，从而返回 false

构造指数表部分：先接收 n 和最小原根 g，然后创建二维向量，行数为（n/10+1），列数为 10，,然后依次求解 g 的 0-p-1 幂次模 n 运算的结果，存储在对应位置，最后返回指数表；

据题意，打印指数表时表中值为-1 的部分打印为“-”；

- **运行示例：**//截图




- **其他：**//用于回答可能预留的问题