Heaven's Light is Our Guide

Rajshahi Universiy of Engineering and Technology

Course Code

ECE 2214

Course Title

Numerical Methods and Discrete Mathematics Sessional

Experiment Date: October 14, 2023,
Submission Date: November 4, 2023

Lab Report 6: Finding Chinese remainder theorem & Carmichael number using python

Submitted to	**Submitted by**
Md. Nahiduzzaman	Md. Tajim An Noor
Lecturer	Roll: 2010025
Dept of ECE, Ruet	

Contents

Modular Exponentiation and Binary Arithmetic in Python.

Md Tajim An Noor

1 Introduction

1.1 Chinese Remainder Theorem

The Chinese remainder theorem, named after the Chinese heritage of problems involving systems of linear congruences, states that when the moduli of a system of linear congruences are pairwise relatively prime, there is a unique solution of the system modulo the product of the moduli.

1.2 Carmichael Number

A composite integer n that satisfies the congruence $b^{\text{n-1}} \equiv 1$ (mod n) for all positive integers b with $gcd(b, n) = 1$ is called a Carmichael number.[1]

2 Tools Used

- Python
- VS Code - for running python code
- MacTeX -LaTeXcompiler
- VS Code with LaTeXworkshop extension as a text editor

3 Process

3.1 Code:

3.1.1 Chinese Reminder Theorem

```
def inv(a, m):
    m0 = m
    x0 = 0
    x1 = 1

    if m == 1:
        return 0

    while a > 1:
        q = a // m
        t = m

        m = a % m
        a = t

        t = x0

        x0 = x1 - q * x0

        x1 = t

    if x1 < 0:
        x1 = x1 + m0

    return x1


def findMinX(num, rem, k):
    prod = 1
    for i in range(0, k):
        prod = prod * num[i]

```

```python
    result = 0

    for i in range(0, k):
        pp = prod // num[i]
        result = result + rem[i] * inv(pp, num[i]) * pp

    return result % prod


num = [5, 7]
rem = [1, 3]
k = len(num)

print(num)
print(rem)
print("x is ", findMinX(num, rem, k))
```

3.1.2 Carmichael Number

```python
# finding GCD
def gcd(a, b):
    if a < b:
        return gcd(b, a)
    if a % b == 0:
        return b
    return gcd(b, a % b)


# finding modular exponent
```

```python
def modExpo(x, y, mod):
    if y == 0:
        return 1
    temp = modExpo(x, y // 2, mod) % mod
    temp = (temp * temp) % mod
    if y % 2 == 1:
        temp = (temp * x) % mod
    return temp


# function to find Carmichael number
def carmaNumber(n):
    b = 2
    while b < n:
        if gcd(b, n) == 1:
            if modExpo(b, n - 1, n) != 1:
                return 0
        b = b + 1
    return 1


for i in range(0, 5):
    x = int(input())
    if carmaNumber(x):
        print(str(x) + " is Carmichael")
    else:
        print(str(x) + " is NOT Carmichael")
```

3.2 Output

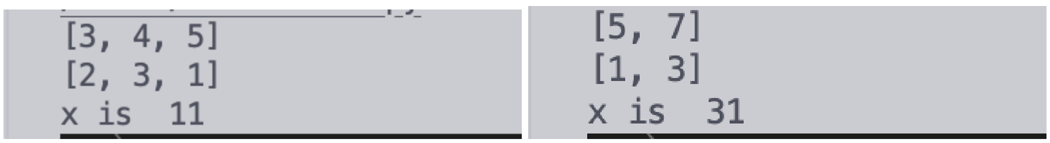

Figure 1: Outputs for Chinese Reminder Theorem

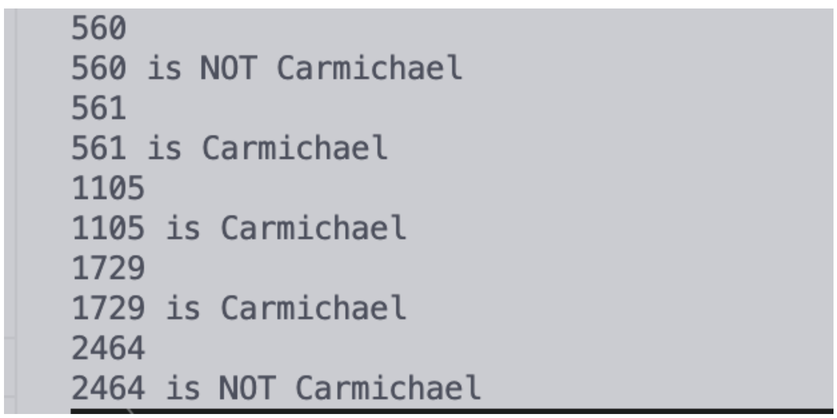

Figure 2: Outputs for Carmichael Number

4 Discussion

In the above codes, for the first one there's another way to find the solution. The one used here uses inverse modulo based implementation. Inputs are the three numbers which are pairwise co-prime, and given remainders of these numbers when an unknown number x is divided by them. [2]
For the Carmichael Number problem, we iterate through all numbers from 1 to n and for every relatively prime number, we check if its $(n-1)^{\text{th}}$ power under modulo n is 1 or not. [2]

References

[1] K. H. Rosen, *DISCRETE MATHEMATICS AND ITS APPLICATIONS,SEVENTH EDITION*. McGraw-Hill.

[2] "Introduction to Chinese Remainder Theorem," Nov. 2022, [Online; accessed 24. Oct. 2023]. [Online]. Available: https://www.geeksforgeeks.org/introduction-to-chinese-remainder-theorem/?ref=lbp