

Other Useful Algorithms

Prime Number

- Sieve of Eratosthenes
 - Initialise set to contain all elements in range 2 - n
 - Sieve starts with smallest element $m = 2$
 - Scan the set and remove all the higher multiples ($2*m$, $3*m$, $4*m$, ... $k*m$)
 - ◆ E.g. remove all multiples of 2, 3, 5 ...
 - Code:

#find all prime numbers from 0 - num

from math import sqrt

num = int(input("enter number: "))

set = [True for i in range(num+1)]

def method1(num):

for m in range(2, int(sqrt(num))+1):

if set[m] == True:

*for n in range(m*m, num+1, m):*

set[n] = False

def method2(num):

p = 2

*while (p*p <= num):*

if set[p] == True:

*for i in range(p*p, num+1, p):*

set[i] = False

p += 1

print("Primes: ", end = "")

for i in range(2, num+1):

if set[i] == True:

print(i, end = " ")

print()

method1(num)

method2(num)

- Increase efficiency:
 - < for j in range (m*m, n+1, m) > instead of < for j in range (m*2, n+1, m) >
 - When multiple of m is checked, m*2, m*3, ... m*(m-1) have been checked
 - Start with m*m instead
 - E.g.:
 - ◆ m = 34
 - ◆ 34*2 sieved when m = 2
 - ◆ 34*3 sieved when m = 3
 - ◆ Start with 34*34, eliminating need to check the multiples before

Check Digit

- Attach weights to the digits
- Sum the product of each weight to the corresponding digit of the code
- Divide the sum using the modulo to find remainder
- Check digit is the difference between modulo and remainder
- Check digit added to the back of the code
- E.g. Modulo 11
 - Weights: 7, 6, 5, 4, 3, 2
 - Code: 508795
 - Modulo: 11
 - Weighted sum: $7 \times 5 + 6 \times 0 + 5 \times 8 + 4 \times 7 + 3 \times 9 + 2 \times 5 = 140$
 - Remainder: $140 / 11 = 12 \text{ R } <8>$
 - Check digit: $11 - 8 = 3$
 - Code: 5087953
- For checking:
 - Find weighted sum of the multiplication of code and weight, check digit has
 - Divide by modulo
 - Weighted sum should be exactly divisible by modulo (no remainder)

- Check digit has weightage of 1
- E.g. Modulo 11
 - ◆ Code = 5087953
 - ◆ Weighted sum = $7 \times 5 + 6 \times 0 + 5 \times 8 + 4 \times 7 + 3 \times 9 + 2 \times 5 + 1 \times 3$
= 143
 - ◆ Remainder = $143 \bmod 11 = 0$
 - ◆ Thus valid code
- Used for small blocks of data

Random Number Generation

- Real random number:
 - All numbers independent of each other
 - All numbers have same probability to occur
- Pseudorandom number:
 - Generator use pre-determined algorithm and begins with seed
 - ◆ Not really random
 - ◆ Pseudorandom
 - Manipulates seed to get sequence of number
 - Sequence of number uniformly distributed in range
 - Code:


```
from random import randint
```

randint(a, b)

inclusive of a and b