1940223_2022-01-18 (divisibility & primality)

March 28, 2022

AIM: Number theory basics

1 Quotient & remainder

1.1 Naive approach

For two integers a and b, to find quotient of a/b, use a//b. To find the remainder of a/b, use a%b.

[23]:
```
a = 257
b = 13
print("(a, b):", (a, b))
print("Quotient:", a//b)
print("Remainder:", a%b)
```

```
(a, b): (257, 13)
Quotient: 19
Remainder: 10
```

1.2 Using divmod

For integers a and b, divmod returns a tuple in the following format: (,)

[24]:
```
a = int(input("a: "))
b = int(input("b: "))
print("(Quotient, Remainder):", divmod(a, b))
```

```
a: 432
b: 42
(Quotient, Remainder): (10, 12)
```

Write a program to input two integers and divides the greater magnitude number by the smaller magnitude number.

[25]:
```
a = int(input("a: "))
b = int(input("b: "))
if abs(a) < abs(b): a, b = b, a
print("Operation: {0} / {1}".format(a, b))
print("(Quotient, Remainder):", divmod(a, b))
```

```
a: 789
b: 56
Operation: 789 / 56
(Quotient, Remainder): (14, 5)
```

2 GCD & LCM

Find the GCD and LCM of two given positive integers.

2.1 Using simple loops

[26]:
```
def gcd(a, b):
    d, i = 1, 1
    while i <= a and i <= b:
        if a % i == 0 and b % i == 0:
            d = i
        i = i + 1
    return d
def lcm(a, b):
    m, i = 1, a*b
    while i >= a and i >= b:
        if i % a == 0 and i % b == 0:
            m = i
        i = i - 1
    return m
```

[27]:
```
a = int(input("a: "))
b = int(input("b: "))
print("(GCD, LCM): ", (gcd(a, b), lcm(a, b)))
```

```
a: 36
b: 24
(GCD, LCM):  (12, 72)
```

2.2 Using Euclidean algorithm

2.2.1 Using loop

[11]:
```
def gcd(a, b):
# Using Euclidean algorithm
    while True:
        # Finding remainder of a / b
        r = a % b
        # If r == 0, return b, else perform above operation on b and r
        if r == 0: return b
        else: a, b = b, r
    """
    ALTERNATE CODE FOR WHILE LOOP:
```

```
    while True:
    if b == 0: return a
    else: a, b = b, a % b
    """
def lcm(a, b):
# Using formula lcm(a, b) = ab / gcd(a, b)
    return int(a * b / gcd(a, b))
```

[12]:
```
a = int(input("a: "))
b = int(input("b: "))
print("(GCD, LCM): ", (gcd(a, b), lcm(a, b)))
```

```
a: 24
b: 36
(GCD, LCM):  (12, 72)
```

2.2.2 Using recursive function

[13]:
```
def gcd(a, b):
    if b == 0: return a
    return gcd(b, a % b)
def lcm(a, b):
# Using formula lcm(a, b) = ab / gcd(a, b)
    return int(a * b / gcd(a, b))
```

[18]:
```
a = int(input("a: "))
b = int(input("b: "))
print("(GCD, LCM): ", (gcd(a, b), lcm(a, b)))
```

```
a: 36
b: 48
(GCD, LCM):  (12, 144)
```

3 Finding divisors of a number

[3]:
```
n = int(input(">> "))
```

```
>> 342
```

The following has a small degree of optimisation, regarding the step value of the iterating variable. Also, I will only find positive divisors, since negative divisors are simply the additive inverses of the positive divisors.

[25]:
```
divisors = []
# If n is odd, i skips even numbers
# If n is even, i runs through all numbers
n = abs(n)
```

```
step = n % 2 + 1
for i in range(1, int(n/2 + 1), step):
    if n % i == 0: divisors.append(i)
# n divides n for n   0...
divisors.append(n)
```

[15]:
```
print(getDivisors(n))
```

```
[1, 2, 3, 6, 9, 18, 19, 38, 57, 114, 171, 342]
```

4 Primes

4.1 Checking if prime

[51]:
```
n = int(input(">> "))
```

```
>> 2321
```

[52]:
```
def isPrime(n):
    if n == 1: return None
    elif n == 0: return False
    else:
        n = abs(n)
        i = 2
        while i*i <= n:
            if n % i == 0:
                i = 0
                break
            i = i + 1
        if i == 0: return False
        else: return True
```

[53]:
```
isPrime(n)
```

[53]:
```
False
```

4.2 Finding primes less than n

[125]:
```
n = abs(int(input(">> ")))
```

```
>> 1242
```

[126]:
```
primes = []
for i in range(2, n):
    # Using the function 'isPrime' defined in the section 'Checking if prime'
    if isPrime(i): primes.append(i)
```

[127]:
```
print(primes)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73,
79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163,
167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251,
257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349,
353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443,
449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557,
563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647,
653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757,
761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863,
877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983,
991, 997, 1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049, 1051, 1061, 1063,
1069, 1087, 1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129, 1151, 1153, 1163,
1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223, 1229, 1231, 1237]
```

4.3 Finding all prime factors of a positive integer

[69]:
```
n = abs(int(input(">> ")))
```

```
>> 32425
```

[70]:
```
def primeFactors(n):
    i, primeFactors = 2, []
    if isPrime(n):
        primeFactors.append(n)
    elif n != 0 or n != 1:
    # Even with only else, the program would produce the same results.
    # However, giving this conditions saves time.
        while i < n/2:
            # Using the function 'isPrime' defined in the section 'Checking if␣
↪prime'
            if n % i == 0 and isPrime(i):
                primeFactors.append(i)
            i = i + 1
    return primeFactors
```

[71]:
```
print(primeFactors(n))
```

```
[5, 1297]
```

4.4 Finding min & max prime factors of a positive integer

I could use the above function and find the minimum and maximum of the list of prime factors. However, this is another approach to this that is less time-consuming, since it does not look for all prime factors, only exactly two. It loops forward once from 2 to n / 2, until it finds a prime factor, then loops backwards once from n / 2 to 2 until it finds a prime factor.

[79]:
```
n = abs(int(input(">> ")))
```

```
>> 342525
```

[81]:
```
minPrimeFactor, maxPrimeFactor = 0, 0

if isPrime(n):
# To save time
    minPrimeFactor, maxPrimeFactor = n, n
elif n != 0 or n != 1:
# Even with only else, the program would produce the same results.
# However, giving this conditions saves time.

    # Finding minimum prime
    i = 2
    while i < n/2:
        # Using the function 'isPrime' defined in the section 'Checking if␣
↪prime'
        if n % i == 0 and isPrime(i):
            minPrimeFactor = i
            break
        i = i + 1

    # Finding maximum prime
    i = int(n / 2)
    while i > 1:
        # Using the function 'isPrime' defined in the section 'Checking if␣
↪prime'
        if n % i == 0 and isPrime(i):
            maxPrimeFactor = i
            break
        i = i - 1
```

[82]:
```
print("Minimum prime factor:", minPrimeFactor)
print("Maximum prime factor:", maxPrimeFactor)
```

```
Minimum prime factor: 3
Maximum prime factor: 4567
```

5 Checking if a number is a multiple of another

[141]:
```
print("Input integers separated by commas...")

# Creating a set to avoid duplicates
# Creating a list to allow iteration
try: N = list(set(map(int, list(input(">> ").split(',')))))
```

```
except: print("Non-integer inputs!")
```

```
Input integers separated by commas…
>> 12,23,34,45,56,67,68,124
```

[142]:
```
isempty = True
for x in range(0, len(N)):
    for y in range(0, len(N)):
        if N[x] != N[y]:
            try:
                if N[y] % N[x] == 0:
                    print("{} is a multiple of {}.".format(N[y], N[x]))
                    isempty = False
            except: pass
if isempty: print("All given integers are coprime.")
```

```
68 is a multiple of 34.
```

6 Perfect numbers

6.1 Check if a number is perfect

[81]:
```
n = abs(int(input(">> ")))
```

```
>> 28
```

[94]:
```
def isPerfect(n):
    sum = 0
    for i in range(1, n//2 + 1):
        if n % i == 0: sum = sum + i
    if sum == n: return True
    else: False
```

[95]:
```
print(isPerfect(n))
```

```
True
```

6.2 Find all perfect numbers in a range

[74]:
```
r = input("Range: ").split(',', 1)
try: r = list(map(int, r))
except: print("Invalid inputs!")
```

```
Range: 1, 10000
```

[75]:
```
perfectNumbers = []
for i in range(r[0], r[1] + 1):
```

```
    if isPerfect(i): perfectNumbers.append(i)
```

[76]:
```
print(perfectNumbers)
```

```
[6, 28, 496, 8128]
```

7 Print all coprime numbers given integer

[2]:
```
def gcd(a, b):
    if b == 0: return a
    return gcd(b, a % b)

def coprimes(n):
    c = []
    for i in range(2, n):
        if gcd(i, n) == 1: c.append(i)
    return c
```

[3]:
```
coprimes(10)
```

[3]:
```
[3, 7, 9]
```