Algorithms - a simple introduction in Python: Part Seven

The Hacker Ethic:
Access to computers - and anything that might teach you something about the way the world works - should be unlimited and total. Always yield to the Hands-On Imperative!
Stephen Levy *Hackers*

Euclid's method for finding the Greatest Common Divisor is perhaps the oldest and most famous algorithm of them all.

The GCD of two integers is the largest integer that divides both numbers with no remainder. So, for instance, the GCD of 14 and 49 is 7. There are many situations in which it is useful to find the GCD of two numbers. For example, when simplifying fractions, GCD will tell you what number to divide the numerator and denominator by.

The algorithm depends upon the following fact: If r is the remainder of a divided by b, then the GCD of a and b is the same as the GCD of b and r.

```
GCD(a, b) = GCD(b, a % b)
```

If we perform this operation repeatedly, we will eventually find that b = 0 and a = the GCD of the original pair of numbers. For instance:

```
GCD(561, 144)  = GCD(144, 129
               = GCD(129, 15)
               = GCD(15, 9)
               = GCD(9, 6)
               = GCD(6, 3)
               = GCD(3, 0)
               = 3
```

Here's the program:

```
# gcd.py Euclid's algorithm for Greatest Common Divisor

def main():
    print(" *** Greatest Common Divisor ***")
    again = 'y'
    while again == 'y':
        x = int(input("x: "))
        y = int(input("y: "))
        print(gcd(x, y))
        again = input("Again? ")

def gcd(a,b):
    """Takes two integers and returns gcd."""
    print(a, b)
    if b == 0:
        return a
    else:
        return gcd(b, a % b)

if __name__ == "__main__":
    main()
```

Once you have that working as expected, have a look at this program on the next page, which will output the prime factors of a number.

```
# prime_factors.py Finds prime factors of an integer

def main():
    display(factor(int(input("Number: "))))

def factor(n):
    """Takes an integer and returns a list of prime factors."""
    i = 2
    factors = []
    while n != 1:
        while n % i == 0:
            n = n // i
            factors.append(i)
        i += 1
    return factors

def display(some_list):
    """Takes a list of factors and prints to the screen."""
    print(some_list[0], end = "")
    if len(some_list) > 1:
        for j in some_list[1:]:
            print(" * " + str(j), end = "")
    else:
        print()

# main
if __name__ == "__main__":
    main()
```

Task

Let's say that instead of the output this program gives:

```
Number: 3628800
2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 3 * 3 * 3 * 3 * 5 * 5 * 7
```

we would prefer for each prime factor to be stored and printed only once.

Modify the program so that it behaves as follows:

```
Number: 3628800
Prime Factors: 2, 3, 5, 7
```

By the way, a prime factors program gives you a simple way to check if a number is prime. We'll be coming back to the primes in later installments.