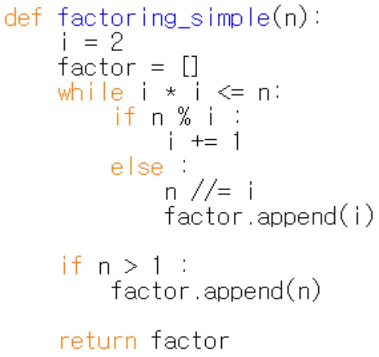
Number Theory Assignment #3

21011712 박준영

I primarily state that the 'math' library was indeed imported, as it does not show in the pictures below.

1) factoring_simple( )

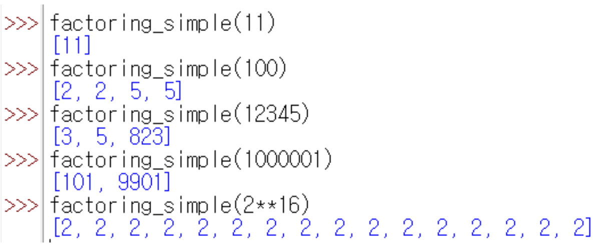


```
def factoring_simple(n):
    i = 2
    factor = []
    while i * i <= n:
        if n % i :
            i += 1
        else :
            n //= i
            factor.append(i)

    if n > 1 :
        factor.append(n)

    return factor
```

the source code



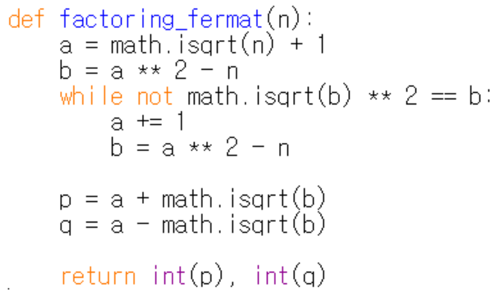
```
>>> factoring_simple(11)
    [11]
>>> factoring_simple(100)
    [2, 2, 5, 5]
>>> factoring_simple(12345)
    [3, 5, 823]
>>> factoring_simple(1000001)
    [101, 9901]
>>> factoring_simple(2**16)
    [2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
```

results of given examples

I have first declared an integer variable I which is going to be used as the dividing number. Then I've declared an array variable called *factor* which will be used as an array to store all the factors. Using the fact that an integer *n* always has a prime divisor smaller than the square root of *n*, I have restricted the condition of the while loop to only run until the square of *i* is smaller than or equal to *n*. If *i* cannot divide *n*, *i* is increased by 1. If *i* can divide *n*, *n* is now divided by that *i*, and that *i* is added to the array. If the square of *i* is greater than *n*, than the while loop ends, adding the remaining value of *n* to the array.

The picture below shows the results of the given examples, and it seems the code is working properly.

2) factoring_fermat( )

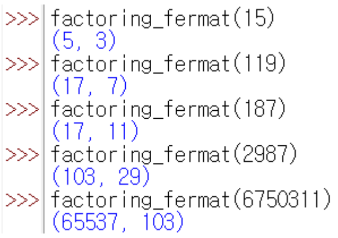


```
def factoring_fermat(n):
    a = math.isqrt(n) + 1
    b = a ** 2 - n
    while not math.isqrt(b) ** 2 == b:
        a += 1
        b = a ** 2 - n

    p = a + math.isqrt(b)
    q = a - math.isqrt(b)

    return int(p), int(q)
```

the source code



```
>>> factoring_fermat(15)
(5, 3)
>>> factoring_fermat(119)
(17, 7)
>>> factoring_fermat(187)
(17, 11)
>>> factoring_fermat(2987)
(103, 29)
>>> factoring_fermat(6750311)
(65537, 103)
```

results of given examples

Since we have to find the value of *square x* and *square y*, I've set the integer variable *a* to the integer value that just exceeds the value of square root of *n*. The integer *b* is naturally set to the value of *square a – n.* If the square of *b* does not match the actual integer *b*, the value of a is added by 1 and the value of *b* is updated according to the new value of *a* as the while loop continues; it means we have yet to find the right values. When the while loop is finished it means we have found the right values. I've set the integer variable *p* as the bigger integer, adding the square root of *b* (which is an integer), and the smaller integer *q* is the value of subtracting the square root of *b* from *a*.

The picture below shows the results of the given examples, and it seems the code is working properly.