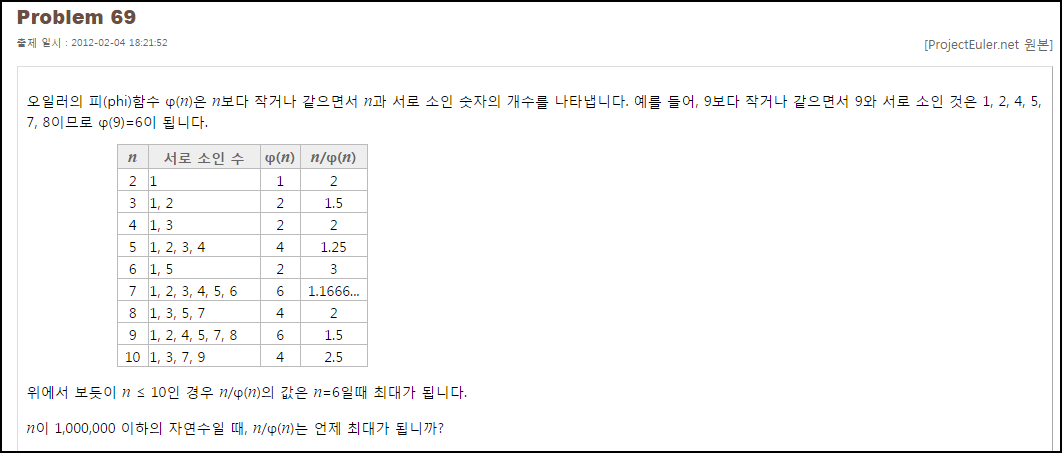
**Exercise**

오일러의 Phi 함수에서 100만 이하의 자연수 중 n/phi(n) 이 가장 큰 값을 가지는 경우에 대해서 구하는 문제입니다.



**Solution\_code**

개념적인 부분에 대해서 짚고, 접근한다면 좀 더 쉬운 해결을 이룰 수 있습니다. 먼저

φ(n) 의 정의를 살펴보면, n 보다 작거나 같으면서 n과 서로소인 숫자의 갯수를 의미합니다.

서로소란 최대공약수 중 가장 큰 약수가 1인것, 즉 최대공약수가 1인 것을 의미합니다. 정의 그대로 옮겨서, 최대공약수를 구하는 함수를 만들고, 아래와 같이 구현을 한 뒤 실행 해보았으나,, 상당히 비효율적이었습니다.

|  |  |  |
| --- | --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14** | **def gcd(a, b):**  **while (b != 0):**  **temp = a % b**  **a = b**  **b = temp**  **return abs(a)**    **def phi(n):**  **cnt = 0**  **for i in range(1,n):**  **if gcd(i,n) == 1:**  **cnt += 1**  **return cnt** | [cs](http://colorscripter.com/info#e) |

그 다음으로 수행해본 것이 , φ(n) 공식을 이용하는 것이었습니다. n의 약수를 a,b 라고 할 때에, φ(n) = n\*(1-1/a)(1-1/b) 이 성립함을 이용하는 방법입니다. 해당 공식을 이용해서 작성해 보았습니다. prime 모듈은 소인수 분해를 위한 모듈로 사용하였으며, fractions 모듈은 분수 계산을 위해서 사용하였습니다.

|  |  |  |
| --- | --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26** | **from prime import \***  **import fractions**  **import time**  **std\_time = time.time()**    **def phi(n):**  **result = 1**  **l = list(set(Prime.factorize(n)))**  **for i in l:**  **result \*= (1-fractions.Fraction(1,i))**  **result \*= n**  **return result**    **n = 2**  **Temp = 1**  **cnt = 0**    **for n in range(2,10\*\*6):**  **result = phi(n)**  **result = float(n)/float(result)**  **# print result**  **if Temp < result:**  **Temp = result**  **cnt = n**  **print '[+]',cnt, Temp, time.time()-std\_time**    ***[Colored by Color Scripter](http://colorscripter.com/info#e)*** | [cs](http://colorscripter.com/info#e) |

**Result**

결과가 썩 마음에 들지는 않지만.. 좀 더 시간을 줄이는 방법이 생각이 나질 않습니다… 다른 분 들 풀이를 보니, 결과적으로 n/φ(n) 는 항상 1보다 크며, 소수인 결론이 나오는 것을 이용 곱이 100만 이하가 될 때까지 곱하면 얼마가 되는지를 이용해서 해결한 신박한 방법도 있었습니다..

