**Số x nhỏ nhất có đúng N ước: x = M(N)**

**2023**

### Số nhỏ nhất có N ước

*Cho số nguyên dương N. Nhiệm vụ của bạn là tìm số K nhỏ nhất, sao cho K có đúng N ước số. Input đảm bảo rằng đáp án không vượt quá*Hiểu đề

Trong toán học, hàm *τ*(*x*) (đọc là tau) cho ra số ước của số nguyên dương x.

Ví dụ,

τ(6) = 4, vì 6 là số nhỏ nhất có 4 ước là 1, 2, 3 và 6.

τ(12) = 6, vì 12 là số nhỏ nhất có 6 ước là 1, 2, 3, 4, 6 và 12.

Lưu ý rằng có nhiều số cùng có số ước như nhau, ví dụ, mọi số nguyên tố p đều có τ(p) = 2. Đề bài yêu cầu: cho N, tìm min K để τ(K) = N.

Trước hết chúng ta làm quen với vài tính chất của hàm τ.

T1. p là số nguyên tố khi và chỉ khi τ(p) = 2.

T2. Nếu x là số *chính phương*, tức là số bằng bình phương của số a nào đó, thì τ(x) là một số lẻ, nếu x không phải là số chính phương thì τ(x) là một số chẵn. Ví dụ, τ(4) = 3; τ(9) = 3; τ(5) = 2; τ(15) = 4.

T3. Số 1 là số duy nhất có τ(1) = 1; các số còn lại đều có τ(x) ≥ 2.

T4. Hàm τ là hàm *nhân*, tức là nếu a và b là hai số nguyên tố cùng nau (hai số có ước chung lớn nhất là 1), thì τ(ab) = τ(a)τ(b). Ví dụ, τ(15) = τ(3)τ(5) = 2⋅2 = 4.

T5. Nếu *x* được phân tích ra thừa số nguyên tố,

thì

Ví dụ, .

Tính chất T5 cho biết tính không đơn trị của hàm τ: có nhiều số khác nhau cho cùng giá trị τ, ví dụ, .

Đây là điểm khó cho bài này.

T6. Nếu N là số nguyên tố thì theo tính chất T5, số K nhỏ nhất có τ(K) = N là

Ví dụ, N = 7, τ(64) = 7.

T7. Giả sử số nguyên dương x là tích của m số không nhất thiết khác nhau nhưng được sắp giảm dần:

Kí hiệu A(x) là số nguyên dương nhỏ nhất có x ước và gọi A(x) là *số thường*. Khi đó hầu hết các số thường đều có dạng

trong đó 2, 3,...,q là dãy m số nguyên tố đầu tiên [1].

Khi đó bài toán được phát biểu như sau:

Cho N, tìm số thường K =A(N)?

Ví dụ, N = 24 = 4⋅3⋅2, .

Theo tính chất T6, khi N nguyên tố thì .

T8. Ta xét một số trường hợp [1][2]

(8.1) Nếu N là tích của 2 số nguyên tố pq được sắp giảm dần (không nhất thiết khác nhau), p ≥ q

thì .

Ví dụ

. Vậy 36 là số nhỏ nhất có 9 ước: τ(36) = 9.

. Vậy 144 là số nhỏ nhất có 15 ước: τ(144) = 15.

(8.2) Nếu N là tích của 3 số nguyên tố pqr được sắp giảm dần (không nhất thiết khác nhau) N = pqr, p ≥ q ≥ r thì , ngoại trừ một trường hợp (\*) sau đây:

* N = 8 = 2⋅2⋅2 đáng lẽ A(8) = 2⋅3⋅5 = 30 nhưng chỉ cần K = 24. Vậy 24 là số nhỏ nhất có 8 ước: τ(24) = 8 (\*).

(8.3) Nếu N là tích của 4 số nguyên tố pqrs được sắp giảm dần (không nhất thiết khác nhau) N = pqrs, p ≥ q ≥ r ≥ s thì , ngoại trừ hai trường hợp (\*) và (\*\*) sau đây:

* N = 16 = 2⋅2⋅2⋅2, đáng lẽ A(16) = 2⋅3⋅5⋅7 = 210 , nhưng chỉ cần = 120. Vậy 120 là số nhỏ nhất có 16 ước: τ(120) = 16 (\*).
* N = 24 = 3⋅2⋅2⋅2 = 24, đáng lẽ nhưng chỉ cần . Vậy 360 là số nhỏ nhất có 24 ước: τ(360) = 24 (\*\*).

(8.4) Nếu N là tích của 5 hoặc 6 số nguyên tố không nhất thiết khác nhau thì có nhiều ngoại lệ. Trường hợp này chúng ta phải Test, cụ thể là sau khi tìm được giá trị K, bạn phải giảm dần K để tìm Kmin. Bạn đọc có thể tham khảo [1].

Ví dụ

N = 32 = 2⋅2⋅2⋅2⋅2, đáng lẽ A(N) = 2⋅3⋅5⋅7⋅11 = 2310, nhưng chỉ cần K = 2⋅2⋅2⋅3⋅5⋅7 = 840. Vậy 840 là số nhỏ nhất có 32 ước: τ(840) = 32.

**Tham khảo**

[1] R. Brown, The minimal number with a given number of divisors, Journal of Number Theory 116 (2006) 150-158.

[2] M. E. Grost. The Smallest Number with a Given Number of Divisors. The American Mathematical Monthly, September 1968, pp. 725-729.

#### Algorithm

Xét số N:

Nếu N là số nguyên tố thì Theo đề bài thì A(N) là số nguyên có tối đa 19 chữ số thập phân, vậy A(N) sẽ là số 64 bit, do đó bạn có thể chọn N = 60.

Chương trình của bạn có nhiệm vụ tính A(N) với 1 ≤ N ≤ 60.

Bạn cần gọi hàm Decomposite để phân tích N thành tích m số nguyên tố p[1..m], cụ thể, hàm Decomp(N) sẽ cho ra dãy m số nguyên tố giảm dần:

N = p[1]⋅p[2]⋅...⋅p[m]

Nếu m = 1, bạn có N = p[1] và do đó N là số nguyên tố, bạn cho ra kết quả

Nếu N là hợp số và m = 2, vận dụng trường hợp (8.1).

Nếu m = 3, vận dụng trường hợp (8.2) chú ý đến ngoại lệ.

Nếu m = 4, vận dụng trường hợp (8.3) chú ý đến ngoại lệ.

Nếu m = 5, vận dụng trường hợp (8.4) chú ý đến ngoại lệ.

Để tính hàm Tau(x) bạn xét các ước u của x tính từ 1 đến .

Khi u =1 ta có ngay hai ước là 1 và x,

Khi u >1 ta có hai ước là u và v, nếu x = u⋅v và u ≠ v.

Nếu ⋅ = *x* thì bớt đi một ước.

#### C++ program

// Tau

#include <bits/stdc++.h>

using namespace std;

typedef unsigned long long UL;

const int MN = 60;

const int pp[] = {0, 2, 3, 5, 7, 11, 13, 17, 19};

const int ii = 8;

int m;

int p[MN]; // cac so nguyen to p[1..m]

void Go() {

cout << " ? ";

fflush(stdin);

if (cin.get() == '.') exit(0);

}

int Tau(UL x) {

if (x == 1) return 1; // Tau(1) = 1

int t = 2;

// Tau(x) luon co 2 uoc la 1 va x

int sx = (int)sqrt(x);

for(UL u = 2; u <= sx; ++u)

if ((x % u) == 0) t += 2;

return((UL)sx\*sx == x) ? t-1 : t;

}

// Decomposite

// Phan tich n ra thua so

void Decomp(int n) {

m = 0;

int sn = (int)sqrt(n);

//cout << "\n Decopm " << n << ":";

for (int u = 2; u <= sn; ++u) {

while((n % u) == 0) {

p[++m] = u;

n /= u;

}

}

if (n > 1) p[++m] = n;

// Sap giam p[1..m]

int i = 1, j = m;

int x;

while(i < j) {

x = p[i]; p[i] = p[j]; p[j] = x;

++i; --j;

}

//for (int i = 1; i <= m; ++i) cout << " " << p[i];

}

// tich(pp[i]^(p[i]-1)), i = 1..m

UL PP(int m) {

UL t = 1;

for(int i = 1; i <= m; ++i) {

t \*= (UL)pow((UL)pp[i], p[i]-1);

}

return t;

}

UL Test(int n, UL k) {

UL y, ymin = k;

for (y = k-1; y > 1; --y)

if (Tau(y) == n) ymin = y;

cout << " Test = " << n << " " << k << " -> " << ymin;

return ymin;

}

UL A(int n) {

if (n == 1) return 1;

Decomp(n);

cout << "\n m = " << m;

if (m == 1)

return ((UL)1 << (n-1));

if (m == 2) return PP(m);

if (m == 3) { // A(8) = 24

return (n == 8) ? 24 : PP(m);

}

if (m == 4) { // A(16) = 120, A(24) = 360

if (n == 16) return 120;

if (n == 24) return 360;

return PP(m);

}

if (m == 5) {

return Test(n, PP(m));

}

}

void Run() {

UL k;

for(int N = 1; N <= MN; ++N) {

k = A(N);

cout << "\n So nho nhat co " << N << " uoc la k = " << k

<< " ? Tau(" << k << ") = " << Tau(k);

}

}

main() {

Run();

// ----------------------------

cout << "\n T H E E N D.";

return 0;

}

#### Output

So nho nhat co 1 uoc la k = 1 ? Tau(1) = 1

m = 1

So nho nhat co 2 uoc la k = 2 ? Tau(2) = 2

m = 1

So nho nhat co 3 uoc la k = 4 ? Tau(4) = 3

m = 2

So nho nhat co 4 uoc la k = 6 ? Tau(6) = 4

m = 1

So nho nhat co 5 uoc la k = 16 ? Tau(16) = 5

m = 2

So nho nhat co 6 uoc la k = 12 ? Tau(12) = 6

m = 1

So nho nhat co 7 uoc la k = 64 ? Tau(64) = 7

m = 3

So nho nhat co 8 uoc la k = 24 ? Tau(24) = 8

m = 2

So nho nhat co 9 uoc la k = 36 ? Tau(36) = 9

m = 2

So nho nhat co 10 uoc la k = 48 ? Tau(48) = 10

m = 1

So nho nhat co 11 uoc la k = 1024 ? Tau(1024) = 11

m = 3

So nho nhat co 12 uoc la k = 60 ? Tau(60) = 12

m = 1

So nho nhat co 13 uoc la k = 4096 ? Tau(4096) = 13

m = 2

So nho nhat co 14 uoc la k = 192 ? Tau(192) = 14

m = 2

So nho nhat co 15 uoc la k = 144 ? Tau(144) = 15

m = 4

So nho nhat co 16 uoc la k = 120 ? Tau(120) = 16

m = 1

So nho nhat co 17 uoc la k = 65536 ? Tau(65536) = 17

m = 3

So nho nhat co 18 uoc la k = 180 ? Tau(180) = 18

m = 1

So nho nhat co 19 uoc la k = 262144 ? Tau(262144) = 19

m = 3

So nho nhat co 20 uoc la k = 240 ? Tau(240) = 20

m = 2

So nho nhat co 21 uoc la k = 576 ? Tau(576) = 21

m = 2

So nho nhat co 22 uoc la k = 3072 ? Tau(3072) = 22

m = 1

So nho nhat co 23 uoc la k = 4194304 ? Tau(4194304) = 23

m = 4

So nho nhat co 24 uoc la k = 360 ? Tau(360) = 24

m = 2

So nho nhat co 25 uoc la k = 1296 ? Tau(1296) = 25

m = 2

So nho nhat co 26 uoc la k = 12288 ? Tau(12288) = 26

m = 3

So nho nhat co 27 uoc la k = 900 ? Tau(900) = 27

m = 3

So nho nhat co 28 uoc la k = 960 ? Tau(960) = 28

m = 1

So nho nhat co 29 uoc la k = 268435456 ? Tau(268435456) = 29

m = 3

So nho nhat co 30 uoc la k = 720 ? Tau(720) = 30

m = 1

So nho nhat co 31 uoc la k = 1073741824

? Tau(1073741824) = 31

m = 5 Test = 32 2310 -> 840

So nho nhat co 32 uoc la k = 840 ? Tau(840) = 32

m = 2

So nho nhat co 33 uoc la k = 9216 ? Tau(9216) = 33

m = 2

So nho nhat co 34 uoc la k = 196608 ? Tau(196608) = 34

m = 2

So nho nhat co 35 uoc la k = 5184 ? Tau(5184) = 35

m = 4

So nho nhat co 36 uoc la k = 1260 ? Tau(1260) = 36

m = 1

So nho nhat co 37 uoc la k = 68719476736

? Tau(68719476736) = 37

m = 2

So nho nhat co 38 uoc la k = 786432 ? Tau(786432) = 38

m = 2

So nho nhat co 39 uoc la k = 36864 ? Tau(36864) = 39

m = 4

So nho nhat co 40 uoc la k = 1680 ? Tau(1680) = 40

m = 1

So nho nhat co 41 uoc la k = 1099511627776

? Tau(1099511627776) = 41

m = 3

So nho nhat co 42 uoc la k = 2880 ? Tau(2880) = 42

m = 1

So nho nhat co 43 uoc la k = 4398046511104

? Tau(4398046511104) = 43

m = 3

So nho nhat co 44 uoc la k = 15360 ? Tau(15360) = 44

m = 3

So nho nhat co 45 uoc la k = 3600 ? Tau(3600) = 45

m = 2

So nho nhat co 46 uoc la k = 12582912 ? Tau(12582912) = 46

m = 1

So nho nhat co 47 uoc la k = 70368744177664

? Tau(70368744177664) = 47

m = 5 Test = 48 4620 -> 2520

So nho nhat co 48 uoc la k = 2520 ? Tau(2520) = 48

m = 2

So nho nhat co 49 uoc la k = 46656 ? Tau(46656) = 49

m = 3

So nho nhat co 50 uoc la k = 6480 ? Tau(6480) = 50

m = 2

So nho nhat co 51 uoc la k = 589824 ? Tau(589824) = 51

m = 3

So nho nhat co 52 uoc la k = 61440 ? Tau(61440) = 52

m = 1

So nho nhat co 53 uoc la k = 4503599627370496

? Tau(4503599627370496) = 53

m = 4

So nho nhat co 54 uoc la k = 6300 ? Tau(6300) = 54

m = 2

So nho nhat co 55 uoc la k = 82944 ? Tau(82944) = 55

m = 4

So nho nhat co 56 uoc la k = 6720 ? Tau(6720) = 56

m = 2

So nho nhat co 57 uoc la k = 2359296 ? Tau(2359296) = 57

m = 2

So nho nhat co 58 uoc la k = 805306368

? Tau(805306368) = 58

m = 1

So nho nhat co 59 uoc la k = 288230376151711744

? Tau(288230376151711744) = 59

m = 4

So nho nhat co 60 uoc la k = 5040 ? Tau(5040) = 60

T H E E N D.

#### Python program

# Tau

# So K nho nhat: Tau(K) = N.

MN = 60

pp = [0, 2, 3, 5, 7, 11, 13, 17, 19]

p = []

m = k = 0

def Go():

if input(' ? ') == '.':

exit(0)

def Tau(x):

if x == 1:

return 1

cx = int(x\*\*0.5)

t = 2 # 1 va x

for u in range(2, cx+1):

if (x % u) == 0:

t += 2

return t-1 if cx\*cx == x else t

def Decomp(x):

global m, p

cx, m = int(x\*\*0.5), 0

p = []

for u in range(2, cx+1):

while (x % u) == 0:

x //= u

p.append(u)

if x > 1:

p.append(x)

p.append(0)

p.reverse()

m = len(p)-1

def PP(m):

t = 1

for i in range(1, m+1):

t \*= pp[i]\*\*(p[i]-1)

return t

# Giam k tim Tau(k) = n

def Test(n, k):

kmin = k

for v in range(k-1, 1, -1):

if Tau(v) == n:

kmin = v

print('Test:', n, k, kmin)

return kmin

# So k nho nhat co n uoc

def A(n):

if n == 1:

return 1

Decomp(n)

# print(n, m, p[1:m+1])

if m == 1: # n nguyen to

return 1 << (n-1)

if m == 2:

return PP(m)

if m == 3:

return 24 if n == 8 else PP(m)

if m == 4:

if n == 16:

return 120

if n == 24:

return 360

return PP(m)

if m == 5:

return Test(n, PP(m))

def Run():

for n in range(1, MN+1):

k = A(n)

print(' So nho nhat co', n, ' uoc la k =',k, end=' ')

print(' ? Tau(',k, ') =', Tau(k))

Run()

print('\n T h e E n d .')

[A005179](http://oeis.org/A005179) lists some resources on the problem. In particular:

[Grost, M. (1968). The Smallest Number with a Given Number of Divisors. The American Mathematical Monthly, 75(7), 725-729. doi:10.2307/2315183](http://www.jstor.org/stable/2315183)

From the introduction:

Given h=q1q2…qnℎ=�1�2…��, with primes q1≥q2≥⋯≥qn�1≥�2≥⋯≥��, let A(h)�(ℎ) be the smallest number with hℎ divisors.

In many cases,

A(h)=2q1−13q2−1…pqn−1n�(ℎ)=2�1−13�2−1…����−1

The primary objective of the paper is to determine the exceptions.

We call these numbers *ordinary*, from [R. Brown, The minimal number with a given number of divisors, Journal of Number Theory 116 (2006) 150-158.](https://www.sciencedirect.com/science/article/pii/S0022314X05001009) In this paper Brown shows almost all A(h)�(ℎ) are ordinary, in particular "We show here that all square-free numbers are ordinary and that the set of ordinary numbers has natural density one."

n lowest number with n factors

1 1

2 2

3 4

4 6

5 16

6 12

7 64

8 24

9 36

10 48

11 1024

12 60

13 4096

14 192

15 144

16 120

17 65536

18 180

19 262144

20 240

Hàm Tau

Run1(): Tìm số nỏ nhất có đúng 240 ước

Run(): Tìm số nỏ nhất có trên 240 ước

Nếu N nguyên tố thì M(N) = 2^(N-1)

Nếu N được phân tích ra thừa số nguyên tố với các mũ = 1 thì

N = p1\*p2\*…\*pk, các p khác nhau và săp giảm

M(N) = q1^(p1-1)\*q2^(p2-1)\*…\*(qk^(pk-1), q sắp tăng

qi là dayc các số nguyên tố đầu tiên: 2, 3, 5, …

M(15) = M(5×3) = 2^4×3^2 = 16×9 = 144

N odd -> x la so chinh phuong: x = a\*a

Example

|  |  |  |  |
| --- | --- | --- | --- |
|  | N=Tau(x) | x |  |
|  | 240  432  448  480  504  512  576  600 | 720720  7207200  8648640  10810800  14414400  17297280  21621600  32432400 | 5.3.2.2.2.2  6.3.3.2.2.2 |
|  | 1000 | 810810000 | 2^4 \* 3^4 \* 5^4 \* 7 \* 11 \* 13 |
|  |  |  |  |
|  | 300 | 2494800 |  |
|  | 400 | 6486480 |  |
|  |  |  |  |
|  |  |  |  |

x N

1. 1

2. 2

3. 4

4. 6

5. 16

6. 12

7. 64

8. 24

9. 36

10. 48

11. 1024

12. 60

13. 4096

14. 192

15. 144

16. 120

17. 65536

18. 180

19. 262144

20. 240

21. 576

22. 3072

24. 360

25. 1296

26. 12288

27. 900

28. 960

30. 720

32. 840

33. 9216

34. 196608

35. 5184

36. 1260

39. 36864

40. 1680

42. 2880

44. 15360

45. 3600

48. 2520

49. 46656

50. 6480

51. 589824

52. 61440

54. 6300

55. 82944

56. 6720

60. 5040

63. 14400

64. 7560

65. 331776

66. 46080

70. 25920

72. 10080

75. 32400

78. 184320

80. 15120

81. 44100

84. 20160

88. 107520

90. 25200

96. 27720

98. 233280

99. 230400

100. 45360

104. 430080

105. 129600

108. 50400

110. 414720

112. 60480

120. 55440

126. 100800

128. 83160

132. 322560

135. 176400

140. 181440

144. 110880

150. 226800

160. 166320

162. 352800

168. 221760

180. 277200

189. 705600

192. 332640

200. 498960

216. 554400

224. 665280

240. 720720

Prog

/\*

Name: Nuoc.CPP

https://www.cplusplus.com/reference

Copyright: (C)2022

Author: Devcpp Fan

Date: 19/03/22 15:45

Description:

So nho nhat co N uoc

Neu N nguyen to thi x = 2^(N-1)

Neu N = p1\*...\*pk (tich cac so ng to khac nhau, tang dan)

thi x = q1^(p1-1)\*...\*qk(p^k-1)

q la day k so nguyen to dau tien

\*/

#include <bits/stdc++.h>

using namespace std;

const int MN = 1001;

typedef long long Long;

int a[MN];

int Tau(Long x) {

if (x == 1) return 1;

if (x == 2) return 2;

// x > 2

int sx = (int)sqrt(x);

int t = 2;

for(int u = 2; u <= sx; ++u) {

if ((x % u) == 0) t += 2;

}

return (sx\*sx == x) ? t-1 : t;

}

void Run1() {

memset(a, 0, sizeof(a));

a[1] = 1; a[2] = 2;

int t;

for (int x = 3; x < 720721; ++x) {

t = Tau(x);

if (a[t] == 0) a[t] = x;

}

for (int i = 1; i <= 240; ++i)

cout << "\n " << i << ". " << a[i];

}

void Run() {

int x = 6486480;

int uoc = 400;

int t;

while(true) {

++x;

t = Tau(x);

if (t > uoc) {

cout << "\n " << x << " " << t;

uoc = t;

}

}

}

void MinTau(int n) {

for (Long x = 810810000-1; x > 32432400; --x)

if (Tau(x) == n) {

cout << x;

return;

}

}

int main () {

// cout << "\n " << Tau(810810000);

// cout << "\n " << Tau(3056130000);

MinTau(1000);

cout << "\n T h e E n d";

return 0;

}