**THỪA SỐ NGUYÊN TỐ**

1. **Prime Facstor**s. Cho số nguyên dương N. Hãy đưa ra tất cả các ước số nguyên tố của N.

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào T bộ test. Mỗi bộ test là một số nguyên dương N được ghi trên một dòng.
* T, N thỏa mãn ràng buộc: 1≤T≤100; 2≤N≤1010.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  315  31 | 3 3 5 7  31 |

1. **Max Prime Facstor**s. Cho số nguyên dương N. Hãy đưa ra ước số nguyên tố lớn nhất của N.

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào T bộ test. Mỗi bộ test là một số nguyên dương N được ghi trên một dòng.
* T, N thỏa mãn ràng buộc: 1≤T≤100; 2≤N≤1010.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  315  31 | 7  31 |

1. **Prime Eratosthenes**. Cho số nguyên dương N. Hãy đưa ra tất cả các số nguyên tố nhỏ hơn hoặc bằng N.

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào T bộ test. Mỗi bộ test là một số nguyên dương N được ghi trên một dòng.
* T, N thỏa mãn ràng buộc: 1≤T≤100; 2≤N≤104.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  10  35 | 2 3 5 7 2 3 5 7 11 13 17 19 23 29 31 |

1. **Least Prime Factor**. Cho số tự nhiên N. Nhiệm vụ của bạn là in ra ước số nguyên tố nhỏ nhất của các số từ 1 đến N. Ước số nguyên tố nhỏ nhất của 1 là 1. Ước số nguyên tố nhỏ nhất của các số chẵn là 2. Ước số nguyên tố nhỏ nhất của các số nguyên tố là chính nó.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N được ghi trên một dòng.
* T, N thỏa mãn ràng buộc: 1≤T≤100; 1≤N≤10000.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 6 10 | 1 2 3 2 5 2 1 2 3 2 5 2 7 2 3 2 |

1. **Prime in Range**. Hãy sinh ra tất cả các số nguyên tố trong khoảng [M, N]. Ví dụ M=1, N=10 ta có kết quả 2 3 5 7.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp mỗi dòng đưa vào một bộ test. Mỗi bộ test là bộ đôi M, N được viết cách nhau một vài khoảng trống.
* T, M, N thỏa mãn ràng buộc: 1≤T≤100; 1≤M≤N≤10000; N-M≤10000.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 1 10 3 5 | 2 3 5 7 3 5 |

1. **Pair Primes**. Cho số nguyên dương chẵn N>2. Hãy đưa ra cặp số nguyên tố p, q đầu tiên có tổng đúng bằng N. Ví dụ N = 6 ta có cặp số nguyên tố đầu tiên là 3 + 3 =6.

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số chẵn N.
* T, N thỏa mãn ràng buộc : 1≤T≤100; 4≤N≤10000.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 74 1024 | 2 71 3 1021 |

1. **Sphenic Number**. Số nguyên dương N được gọi là số Sphenic nếu N được phân tích duy nhất dưới dạng tích của ba số khác nhau. Ví dụ N=30 là số Sphenic vì 30 = 2×3×5; N = 60 không phải số Sphenic vì 60 = 2×2×3×5. Cho số tự nhiên N, nhiệm vụ của bạn là kiểm tra xem N có phải số Spheic hay không?

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số nguyên dương N.
* T, N thỏa mãn ràng buộc : 1≤T≤100; 1≤N≤10000.

Output:

* Đưa ra 1 hoặc 0 tương ứng với N là số Sphenic hoặc không của mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 30 60 | 1 0 |

1. Cho số tự nhiên N và số nguyên tố P. Nhiệm vụ của bạn là tìm số x lớn nhất để N! chia hết cho px. Ví dụ với N=7, p=3 thì x=2 là số lớn nhất để 7! Chia hết cho 32.

Input:

* Dòng đầu tiên đưa vào số lượng bộ test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là cặp số N, p được viết cách nhau một vài khoảng trống.
* T, N, p thỏa mãn rang buộc : 1≤T≤100; 1≤N≤105; 2≤p≤5000;

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 3 62 7 76 2 3 5 | 9 73 0 |

1. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy đưa ra tất cả các ước số nguyên tố của N cùng lũy thừa của nó. Ví dụ N=100 = 22 × 52. N = 35 =51 × 71.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số nguyên N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N≤10000.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 100 35 | 2 2 5 2 5 1 7 1 |

1. **Smith Number**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy kiểm tra N có phải là số Smith hay không. Một số được gọi là số Smith nếu N không phải là số nguyên tố và có tổng các chữ số của N bằng tổng các chữ số của các ước số nguyên tố của N. Ví dụ N = 666 có các ước số nguyên tố là 2, 3, 3, 37 có tổng các chữ số là 18.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số nguyên N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N≤100000.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 4 666 | Yes No |

1. **Perfect Number**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy kiểm tra N có phải là số hoàn hảo hay không. Một số N được gọi là số hoàn hảo nếu tổng các ước số của nó bằng chính nó. Ví dụ N = 6=1 + 2 + 3 là số hoàn hảo.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số nguyên N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N≤1018.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 6 21 | 1 0 |

1. **Kth Prime Divisors**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy đưa ra ước số nguyên tố thứ k của N. Đưa ra -1 nếu không tồn tại ước số thứ k của N. Ví dụ N = 255, k =2 ta có kết quả là 3 vì 255 = 3×3×5×5. Với N = 81, k = 5 ta có kết quả -1 vì 81 = 3×3×3×3.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một bộ đôi N và k.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N, k≤104.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 25 5 81 5 | 3 -1 |

1. **Số các số có ước số lẻ**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy đếm số các số có ước số lẻ nhỏ hơn hoặc bằng N.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N ≤109.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2 1 5 | 1 2 |

1. **Number 3-Divisors**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy liệt kê tất cả các số có đúng ba ước số. Ví dụ n=100, ta có các số 4, 9, 25, 49.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N ≤106.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  50  200 | 4 9 25 49  4 9 25 49 121 169 |

1. **Smallest Divisors**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy đưa ra ước số nguyên tố nhỏ nhất của các số từ 1 đến N. Ví dụ n=10, ta có được kết quả : 1 2 3 2 5 2 7 2 3 2 .

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N ≤106.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  5  10 | 1 2 3 2 5  1 2 3 2 5 2 7 2 3 2 |

1. **Number 3-Divisors**. Cho số tự nhiên N. Nhiệm vụ của bạn là hãy đếm tất cả các số có đúng ba ước số. Ví dụ n=100, ta có các số 4.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N ≤1012.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  50  200 | 4  6 |

1. **Number 3-Divisors**. Cho hai số L, R. Nhiệm vụ của bạn là hãy đếm tất cả các số có đúng ba ước số trong khoảng [L, R]. Ví dụ L =1, R =10, ta có kết quả là 2 vì chỉ có số 3 và 9 là có đúng 3 ước số.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là cặp số L, R.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤L, R ≤1012.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  1 10  1 1000000000000 | 4  78498 |

1. **kth-Divisors (new)**. Cho số tự nhiên số N. Nhiệm vụ của bạn là in ra tất cả các ước số của N theo thứ tự tăng dần. Ví dụ N=10 ta có kết quả là 1, 2, 5, 10.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là một số N.
* T, N thỏa mãn rang buộc 1≤T≤100; 1≤N ≤1012.

Output:

* Đưa ra kết quả mỗi test theo từng dòng.

|  |  |
| --- | --- |
| Input: | Output: |
| 2  10  100 | 1 2 5 10  1 2 4 5 10 20 25 50 100 |

1. **kth-Divisors (new)**. Cho hai số N và K. Nhiệm vụ của bạn là đưa ra ước số nhỏ nhất thứ K của N. Ví dụ N=4, K = 2 ta có kết quả là 2 vì 4 có 3 ước số 1, 2, 4. Với N=5, K = 3 ta có kết quả -1 vì N chỉ có 2 ước số là 1 và 5.

Input:

* Dòng đầu tiên đưa vào số lượng test T.
* Những dòng kế tiếp đưa vào các bộ test. Mỗi bộ test là cặp số N, K.
* T, N, K thỏa mãn rang buộc 1≤T≤100; 1≤N ≤1012, K≤106.

Output:

* Đưa ra kết quả mỗi test theo từng dòng. Đưa ra -1 nếu không tồn tại ước số nhỏ nhất thứ k

|  |  |
| --- | --- |
| Input: | Output: |
| 2  4 2  5 3 | 2  -1 |

B1. Prime 1. <https://www.geeksforgeeks.org/print-all-prime-factors-of-a-given-number/>

B3. <https://www.geeksforgeeks.org/sieve-of-eratosthenes/>

B4.

B5. <https://practice.geeksforgeeks.org/problems/find-prime-numbers-in-a-range/0/?ref=self>

B6. <https://practice.geeksforgeeks.org/problems/return-two-prime-numbers/0/?ref=self>

<https://www.geeksforgeeks.org/smith-number/>

<https://www.geeksforgeeks.org/sphenic-number/>

<https://www.geeksforgeeks.org/k-th-prime-factor-given-number/>

<https://www.geeksforgeeks.org/rsa-algorithm-cryptography/>

Mark 8. <https://practice.geeksforgeeks.org/problems/generalised-fibonacci-numbers/0>

<https://practice.geeksforgeeks.org/problems/total-decoding-messages/0>

<https://www.geeksforgeeks.org/count-ways-reach-nth-stair/>

<https://practice.geeksforgeeks.org/problems/fibonacci-sum/0>

<https://www.geeksforgeeks.org/nth-non-fibonacci-number/>

<https://practice.geeksforgeeks.org/problems/nth-even-fibonacci-number/0>

<https://practice.geeksforgeeks.org/problems/even-fibonacci-numbers-sum/0>

Mark 8. <https://practice.geeksforgeeks.org/problems/return-two-prime-numbers/0/?ref=self>

<https://www.geeksforgeeks.org/print-all-prime-factors-and-their-powers/>

<https://www.geeksforgeeks.org/no-factors-n/>

<https://practice.geeksforgeeks.org/problems/prime-factorization-and-geek-number/0/?ref=self>

<https://www.geeksforgeeks.org/find-politeness-number/>

<https://www.geeksforgeeks.org/find-sum-even-factors-number/>

<https://www.geeksforgeeks.org/sum-factors-number/>

<https://www.geeksforgeeks.org/find-minimum-sum-factors-number/>

<https://www.geeksforgeeks.org/print-all-prime-factors-of-a-given-number/>

<https://www.geeksforgeeks.org/find-divisors-natural-number-set-1/>

<https://www.geeksforgeeks.org/find-all-divisors-of-a-natural-number-set-2/>

<https://www.geeksforgeeks.org/find-numbers-n-divisors-given-range/>

<https://www.geeksforgeeks.org/sum-of-all-proper-divisors-of-a-natural-number/>

<https://www.geeksforgeeks.org/sum-divisors-1-n/>

<https://www.geeksforgeeks.org/check-door-open-closed/>

<https://www.geeksforgeeks.org/legendres-formula-highest-power-of-prime-number-that-divides-n/>