The kth Factor of n

1492. The kth Factor of n

Medium 1334 257 Add to List Share

You are given two positive integers `n` and `k`. A factor of an integer `n` is defined as an integer `i` where `n % i == 0`.

Consider a list of all factors of `n` sorted in **ascending order**, return *the* k^{th} *factor* in this list or return `-1` if `n` has less than `k` factors.

Example 1:

```
Input: n = 12, k = 3
Output: 3
Explanation: Factors list is [1, 2, 3, 4, 6, 12], the 3rd factor is 3.
```

Example 2:

```
Input: n = 7, k = 2
Output: 7
Explanation: Factors list is [1, 7], the 2nd factor is 7.
```

Example 3:

```
Input: n = 4, k = 4
Output: -1
Explanation: Factors list is [1, 2, 4], there is only 3 factors. We should return -1.
```

Brute force

n=12

Find all factor of n

loop(i: 0 → n)

if (n%i == 0)

v.push_back(i)

since v is already Sorted

return k^{th} factor

Better approch:-

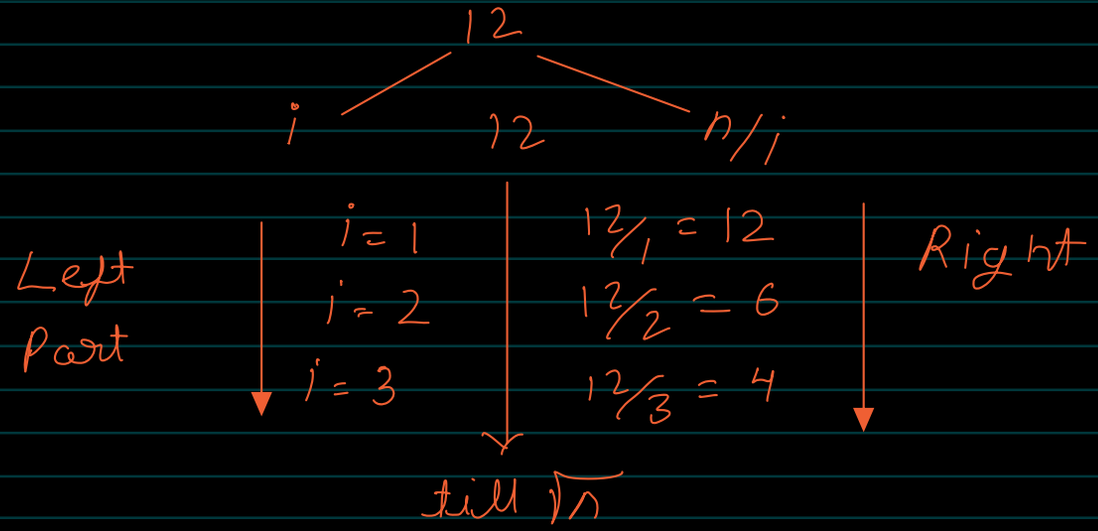
instead of running loop till n we can n/2.

```
class Solution {
public:
    int kthFactor(int n, int k) {
    vector<int> v;
    int temp=k;
    for (int i = 1; i <= n/2 && temp!=v.size(); i++) {
        if (n % i == 0) {
            v.push_back(i);
        }
    }
    v.push_back(n);

    if (k <= v.size()) {
        return v[k-1];
    }
    return -1;
}

};
```

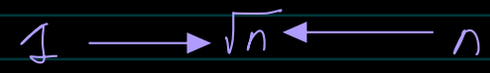
Optimal approach:

$n = 12$ and i is factor of n

then n/i is also factor of 12




```
//Left Part
loop (i: 1 --> i*i<n)
    if ( n%i==0 && --k==0)
        return i
```




```
Right part
  loop (i=Sqrt(n); i>=1; i--)
      if ( n%i==0 && --k==0)
          return n/i;

return -1;
```

Answer depends upon k

if k is 1 then we will get our ans at first condition

i.e $n=12$, $k=3$

```
loop (i:1 ; i*i<=n;)
    if ( 12%1==0 && --k==0
                    2 == 0 False
```

4 < 12

i++

if (12 % 2 == 0 && --k == 0)
True 1 == 0 False

i++ 9 < 12

if (12 % 3 == 0 && --k == 0)
True 0 == 0 True

return i // 3

```
class Solution {
public:
    int kthFactor(int n, int k) {
        int root=sqrt(n);
        for(int i=1;i*i<n;i++){
            if(n%i==0 && --k==0){
                return i;
            }
        }
        for(int i=root;i>=1;i--){
            if(n%i==0 && --k==0){
                return n/i;
            }
        }
        return -1;
    }
};
```

Sqrt(n)

Sqrt(n)

T.C - O(sqrt(n))

S.C - O(1)