

**WEEK:12**

**USER-DEFINED FUNCTIONS &**  
**RECURSIVE FUNCTIONS**

## **WEEK:12-01 [USER DEFINED FUNCTION]**

**ROLL NO: 240801161**

**NAME: KEERTHANA S**

Status	Finished
Started	Monday, 13 January 2025, 3:42 AM
Completed	Monday, 13 January 2025, 4:10 AM
Duration	27 mins 43 secs

### **QUESTION:1**

## **FIND THE FACTOR**

### **Problem Statement:**

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the pth element of the list, sorted ascending. If there is no pth element, return 0.

### **Example**

n = 20

p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

### **Function Description**

Complete the function pthFactor in the editor below.

pthFactor has the following parameter(s):

int n: the integer whose factors are to be found

int p: the index of the factor to be returned

### **Returns:**

int: the long integer value of the pth integer factor of n or, if there is no factor at that index, then 0 is returned

### **Constraints**

$1 \leq n \leq 10^{15}$

$1 \leq p \leq 10^9$

### **Input Format for Custom Testing**

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer  $n$ , the number to factor.

The second line contains an integer  $p$ , the 1-based index of the factor to return.

### **Sample Input**

STDIN Function

-----

10  $\rightarrow$   $n = 10$

3  $\rightarrow$   $p = 3$

### **Sample Output**

5

### **Explanation**

Factoring  $n = 10$  results in  $\{1, 2, 5, 10\}$ . Return the  $p = 3$ rd factor, 5, as the answer.

## PROGRAM:

Answer: (penalty regime: 0 %)

Reset answer

```
1  /*
2  * Complete the 'pthFactor' function below.
3  *
4  * The function is expected to return a LONG_INTEGER.
5  * The function accepts following parameters:
6  * 1. LONG_INTEGER n
7  * 2. LONG_INTEGER p
8  */
9  #include <stdio.h>
10 #include <math.h>
11 long pthFactor(long n, long p)
12 {
13     int count=0;
14     long i;
15     for(i=1;i<=sqrt(n); i++)
16     {
17         if(n%i == 0)
18         {
19             count++;
20             if(count==p)
21             {
22                 return i;
23             }
24         }
25     }
26     if(sqrt(n)*sqrt(n) == n){
27         count--;
28     }
29     for(i=sqrt(n);i>=1;i--)
30     {
31         if(n%i ==0 && i!= n/i)
32         {
33             count++;
34             if(count == p)
35             {
36                 return n/i;
37             }
38         }
39     }
40 }
41 return 0;
42 }
```

## OUTPUT:

	Test	Expected	Got	
✓	printf("%ld", pthFactor(10, 3))	5	5	✓
✓	printf("%ld", pthFactor(10, 5))	0	0	✓
✓	printf("%ld", pthFactor(1, 1))	1	1	✓

Passed all tests! ✓

## **QUESTION:2**

### **4<sup>TH</sup> BIT**

#### **Problem Statement:**

A binary number is a combination of 1s and 0s. Its nth least significant digit is the nth digit starting from the right starting with 1. Given a decimal number, convert it to binary and determine the value of the 4th least significant digit.

#### **Example**

number = 23

- Convert the decimal number 23 to binary number:  $23_{10} = 2^4 + 2^2 + 2^1 + 2^0 = (10111)_2$ .
- The value of the 4th index from the right in the binary representation is 0.

#### **Function Description**

Complete the function fourthBit in the editor below.

fourthBit has the following parameter(s):

int number: a decimal integer

#### **Returns:**

int: an integer 0 or 1 matching the 4th least significant digit in the binary representation of number.

#### **Constraints**

$0 \leq \text{number} < 2^{31}$

#### **Input Format for Custom Testing**

Input from stdin will be processed as follows and passed to the function.

The only line contains an integer, number.

#### **Sample Input**

STDIN Function

-----

32 → number = 32

## Sample Output

0

## Explanation

- Convert the decimal number 32 to binary number:  $32_{10} = (100000)_2$ .
- The value of the 4th index from the right in the binary representation is 0.

## PROGRAM:

```
1  /*  
2  * Complete the 'fourthBit' function below.  
3  *  
4  * The function is expected to return an INTEGER.  
5  * The function accepts INTEGER number as parameter.  
6  */  
7  #include<stdio.h>  
8  
9  int fourthBit(int number)  
10 {  
11  
12     return (number >> 3) & 1;  
13 }  
14  
15  
16  
17  
18  
19  
20  
21  
22
```

## OUTPUT:

	Test	Expected	Got	
✓	printf("%d", fourthBit(32))	0	0	✓
✓	printf("%d", fourthBit(77))	1	1	✓

Passed all tests! ✓