Week-12-01

Name: Mukesh V **Roll No: 241901059**

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 ≤ n ≤ 1015
1 ≤ p ≤ 109

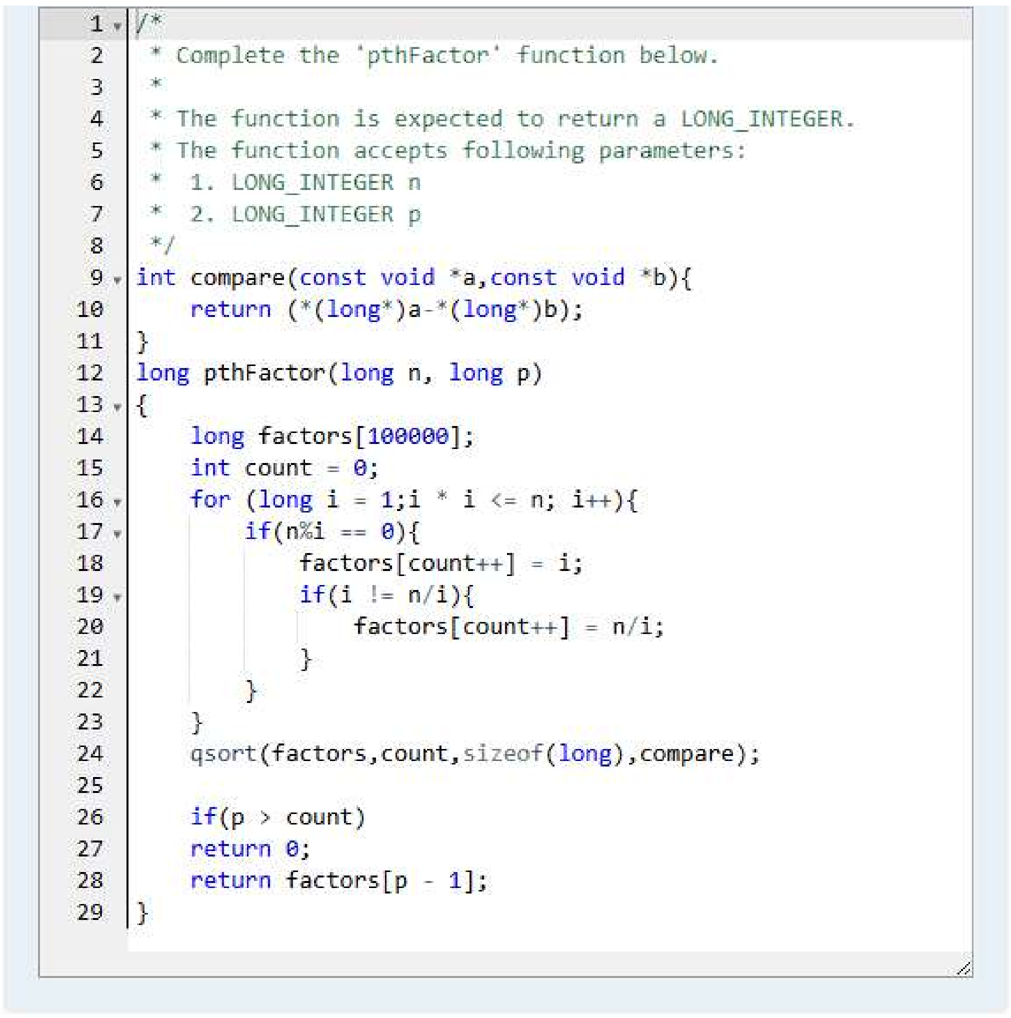
Input Format for Custom Testing
Input from stdin will be processed as follows and passed to the function. The first line contains an integer n, the number to factor.
The second line contains an integer p, the 1-based index of the factor to return.

Sample Input
```
STDIN          Function
-----          -----
10     →       n = 10
3      →       p = 3
```

Sample Output
```
5
```

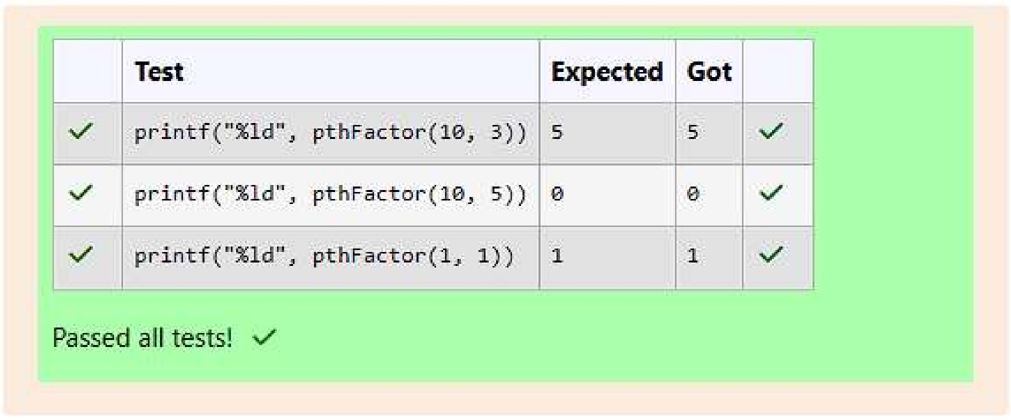
Explanation
Factoring n = 10 results in {1, 2, 5, 10}. Return the p = 3rd factor, 5, as the answer.

Program:



```c
/*
 * Complete the 'pthFactor' function below.
 *
 * The function is expected to return a LONG_INTEGER.
 * The function accepts following parameters:
 *  1. LONG_INTEGER n
 *  2. LONG_INTEGER p
 */
int compare(const void *a,const void *b){
    return (*(long*)a-*(long*)b);
}
long pthFactor(long n, long p)
{
    long factors[100000];
    int count = 0;
    for (long i = 1;i * i <= n; i++){
        if(n%i == 0){
            factors[count++] = i;
            if(i != n/i){
                factors[count++] = n/i;
            }
        }
    }
    qsort(factors,count,sizeof(long),compare);

    if(p > count)
    return 0;
    return factors[p - 1];
}
```



	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! ✓

4th 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 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 ≤ number < 231

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

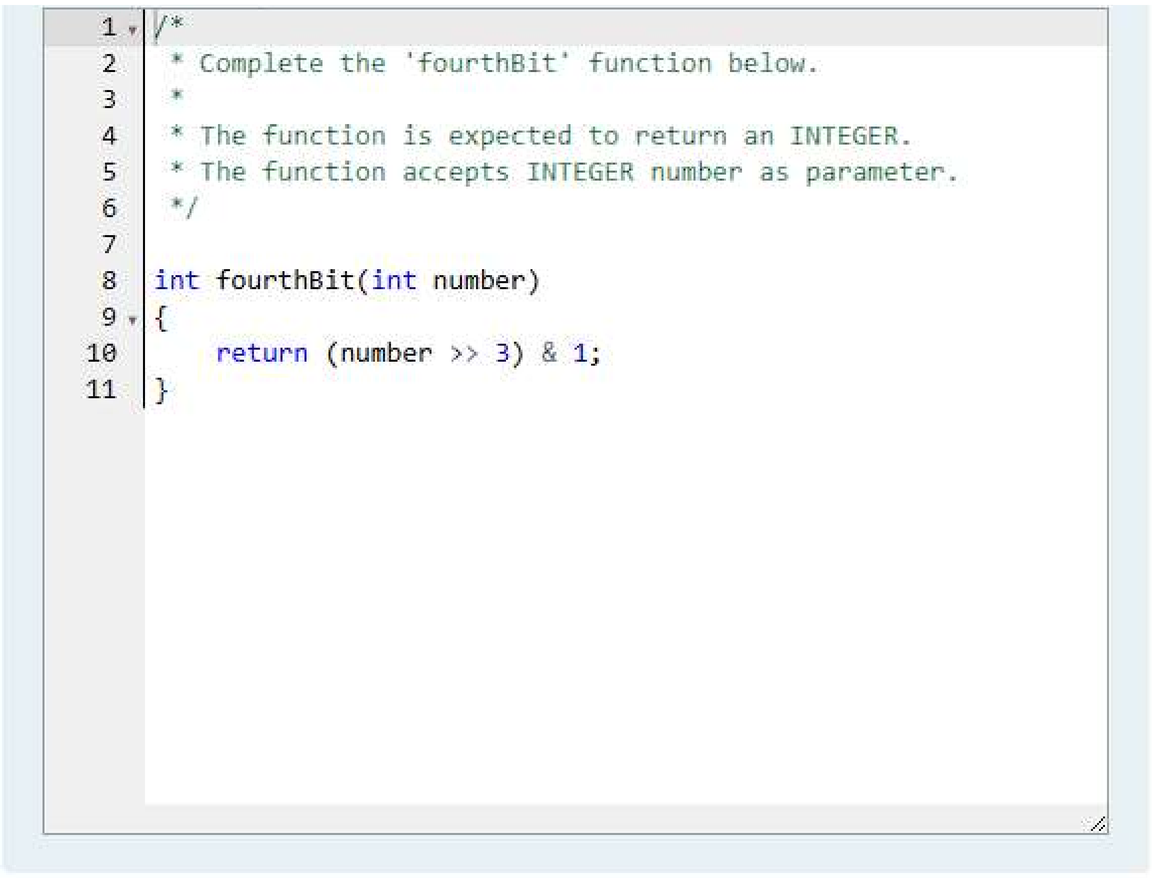
```
STDIN          Function
----           -----
32      →      number = 32
```

Sample Output
0

Explanation

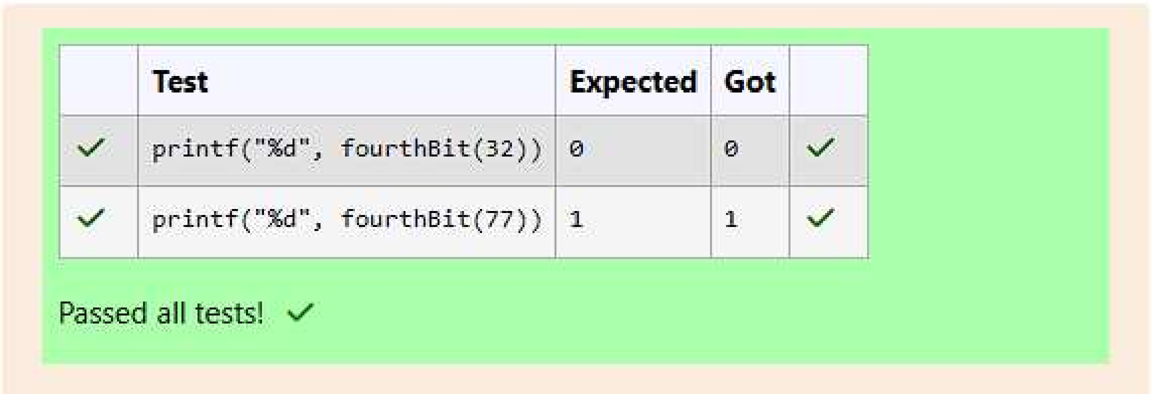
- Convert the decimal number 32 to binary number: 3210 = (100000)2.
- The value of the 4th index from the right in the binary representation is 0.

Program:



```
/*
 * Complete the 'fourthBit' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts INTEGER number as parameter.
 */

int fourthBit(int number)
{
    return (number >> 3) & 1;
}
```



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

Passed all tests! ✓