GE23131-Programming Using C-2024

Quiz navigation

1	2

Show one page at a time

Finish review

Question **1**

Correct

Marked out of 1.00

⚑ Flag question

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.

Returns:

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

Constraints

$0 \le number < 2^{31}$

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 Case 0

Sample Input 0

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

Sample Output 0

Explanation 0

·   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.

Sample Case 1

Sample Input 1

STDIN   Function

-----   --------

77   →   number = 77

Sample Output 1

1

Explanation 1

·   Convert the decimal number 77 to binary number: $77_{10} = (1001101)_2$.

·   The value of the 4th index from the right in the binary representation is 1.

Answer: (penalty regime: 0 %)

```
 1    int fourthBit(int number) {
 2        int binary[32];
 3        int i=0;
 4        while(number>0) {
 5            binary[i]=number%2;
 6            number/=2;
 7            i++;
 8        }
 9        if(i>=4) {
10            return binary[3];
11        } else {
12            return 0;
13        }
14    }
```

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

Passed all tests! ✓

Question **2**

Correct

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 p^{th} element, return 0.

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

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

$1 \le p \le 10^9$

Input Format for Custom Testing

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 Case 0

Sample Input 0

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

Sample Output 0

5

Explanation 0

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

Sample Case 1

Sample Input 1

```
10   →  n = 10
5    →  p = 5
```

Sample Output 1

0

Explanation 1

Factoring n = 10 results in {1, 2, 5, 10}. There are only 4 factors and p = 5, therefore 0 is returned as the answer.

Sample Case 2

Sample Input 2

```
STDIN    Function
-----    --------
1    →  n = 1
1    →  p = 1
```

Sample Output 2

1

Explanation 2

Answer: (penalty regime: 0 %)

Reset answer

```
1   long pthFactor (long n, long p) {
2     int count=0;
3     for(long i=1;i<=n; ++i) {
4       if(n%i==0) {
5         count++;
6         if(count==p) {
7           return i;
8         }
9       }
10    }
11    return 0;
12  }
```

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

Finish review