

# Week-12-Recursive functions

Question **1**

Correct

Marked out of  
1.00[Flag question](#)

You are a bank account hacker. Initially you have 1 rupee in your account, and you want exactly **N** rupees in your account. You wrote two hacks, first hack can multiply the amount of money you own by 10, while the second can multiply it by 20. These hacks can be used any number of time. Can you achieve the desired amount **N** using these hacks.

**Constraints:**

$$1 \leq T \leq 100$$

$$1 \leq N \leq 10^{12}$$

**Input**

- The test case contains a single integer N.

**Output**

For each test case, print a single line containing the string "1" if you can make exactly N rupees or "0" otherwise.

SAMPLE INPUT

1

## Source code

```
1  /*
2  * Complete the 'myFunc' function below.
3  *
4  * The function is expected to return an INTEGER.
5  * The function accepts INTEGER n as parameter.
6  */
7
8  int myFunc(int n)
9  {
10     while(n>1){
11         if(n%20==0){
12             n/=20;
13         }
14         else if(n%10==0)
15         {
16             n/=10;
17         }
18         else
19             return 0;
20     }
21     return(n==1) ? 1:0;
22 }
23
```

## Result

	Test	Expected	Got	
✓	printf("%d", myFunc(1))	1	1	✓
✓	printf("%d", myFunc(2))	0	0	✓
✓	printf("%d", myFunc(10))	1	1	✓
✓	printf("%d", myFunc(25))	0	0	✓
✓	printf("%d", myFunc(200))	1	1	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of  
1.00

🚩 Flag question

Find the number of ways that a given integer,  $X$ , can be expressed as the sum of the  $N^{\text{th}}$  powers of unique, natural numbers.

For example, if  $X = 13$  and  $N = 2$ , we have to find all combinations of unique squares adding up to  $13$ . The only solution is  $2^2 + 3^2$ .

#### Function Description

Complete the powerSum function in the editor below. It should return an integer that represents the number of possible combinations.

powerSum has the following parameter(s):

X: the integer to sum to

N: the integer power to raise numbers to

Input Format

The first line contains an integer  $X$ .

The second line contains an integer  $N$ .

#### Constraints

$$1 \leq X \leq 1000$$

$$2 \leq N \leq 10$$

[Source code](#)

```
1 #include <math.h>
2 /*
3  * Complete the 'powerSum' function below.
4  *
5  * The function is expected to return an INTEGER.
6  * The function accepts following parameters:
7  * 1. INTEGER x
8  * 2. INTEGER n
9  */
10
11 int powerSum(int x, int m, int n)
12 {
13     int power= pow(m,n);
14     if(power>x){
15         return 0;
16     }
17     if(power == x){
18         return 1;
19     }
20
21     return powerSum(x-power, m+1, n)+powerSum(x, m+1, n);
22 }
23
24 int powerSumWrapper(int x, int n){
25     return powerSum(x, 1,n);
26 }
27
```

---

## Result

	Test	Expected	Got	
✓	printf("%d", powerSum(10, 1, 2))	1	1	✓

Passed all tests! ✓



