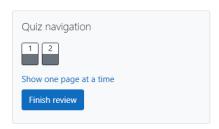
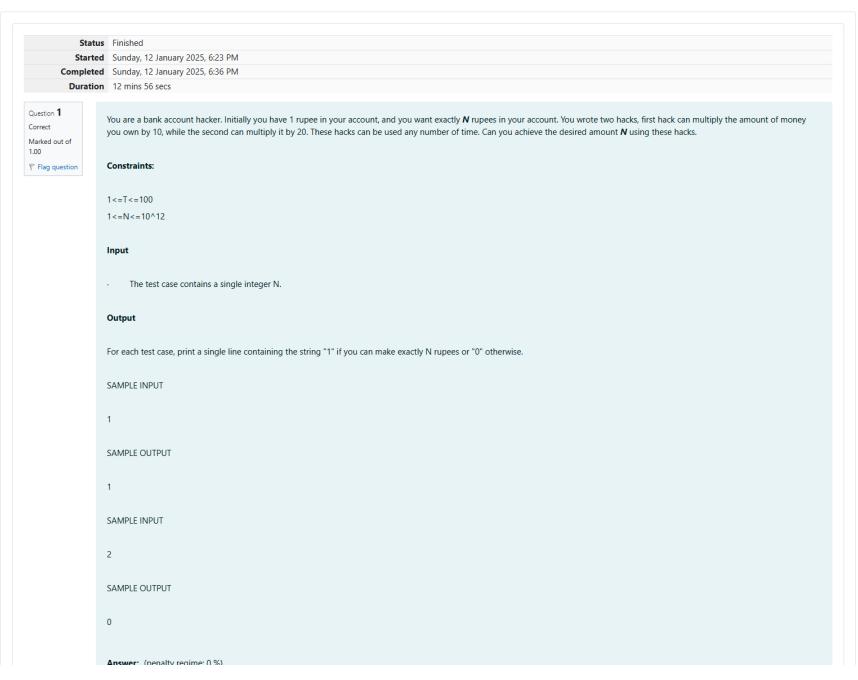
GE23131-Programming Using C-2024





```
Reset answer
       * Complete the 'myFunc' function below.
  3
  4
       \ensuremath{^{*}} The function is expected to return an INTEGER.
   5
       * The function accepts INTEGER n as parameter.
   6
   8
      int myFunc(int n)
   9 ,
  10
         if(n==1)
  11
         return 1;
         if(n%10==0)
  12
         if(myFunc(n/10)==1)
  13
  14
         return 1;
         if(n%20==0)
  15
  16
         if(myFunc(n/20)==1)
  17
          return 1;
  18
          return 0;
  19
  20
```

	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**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.

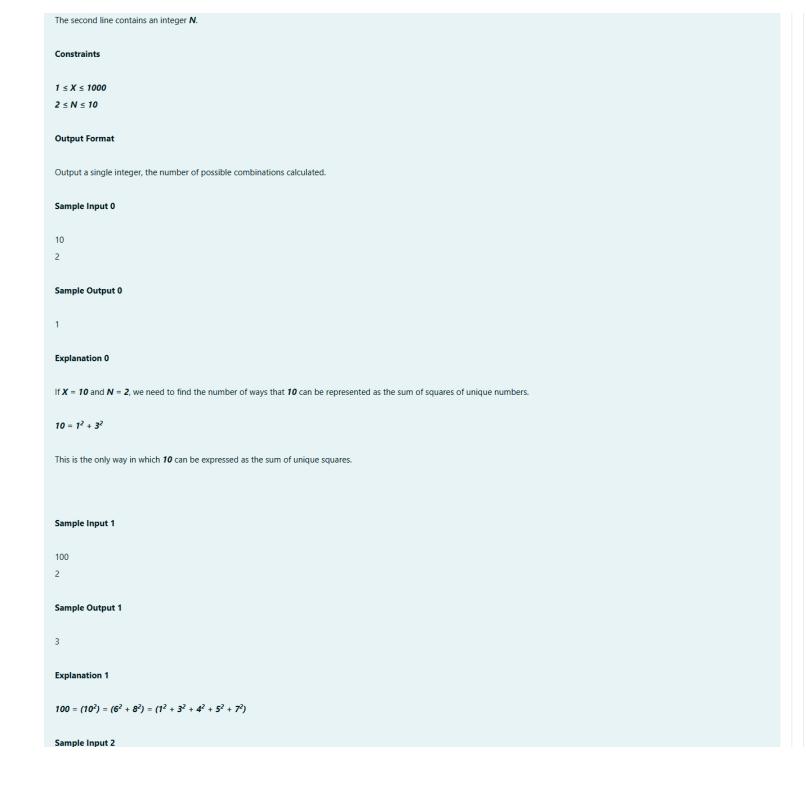
powerSum has the following parameter(s):

X: the integer to sum to

N: the integer power to raise numbers to

Input Format

The first line contains an integer X.



```
100
3
Sample Output 2
1
Explanation 2
100 can be expressed as the sum of the cubes of 1, 2, 3, 4.
(1 + 8 + 27 + 64 = 100). There is no other way to express 100 as the sum of cubes.
Answer: (penalty regime: 0 %)
```

```
Reset answer
```

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

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

Passed all tests! <

Finish review