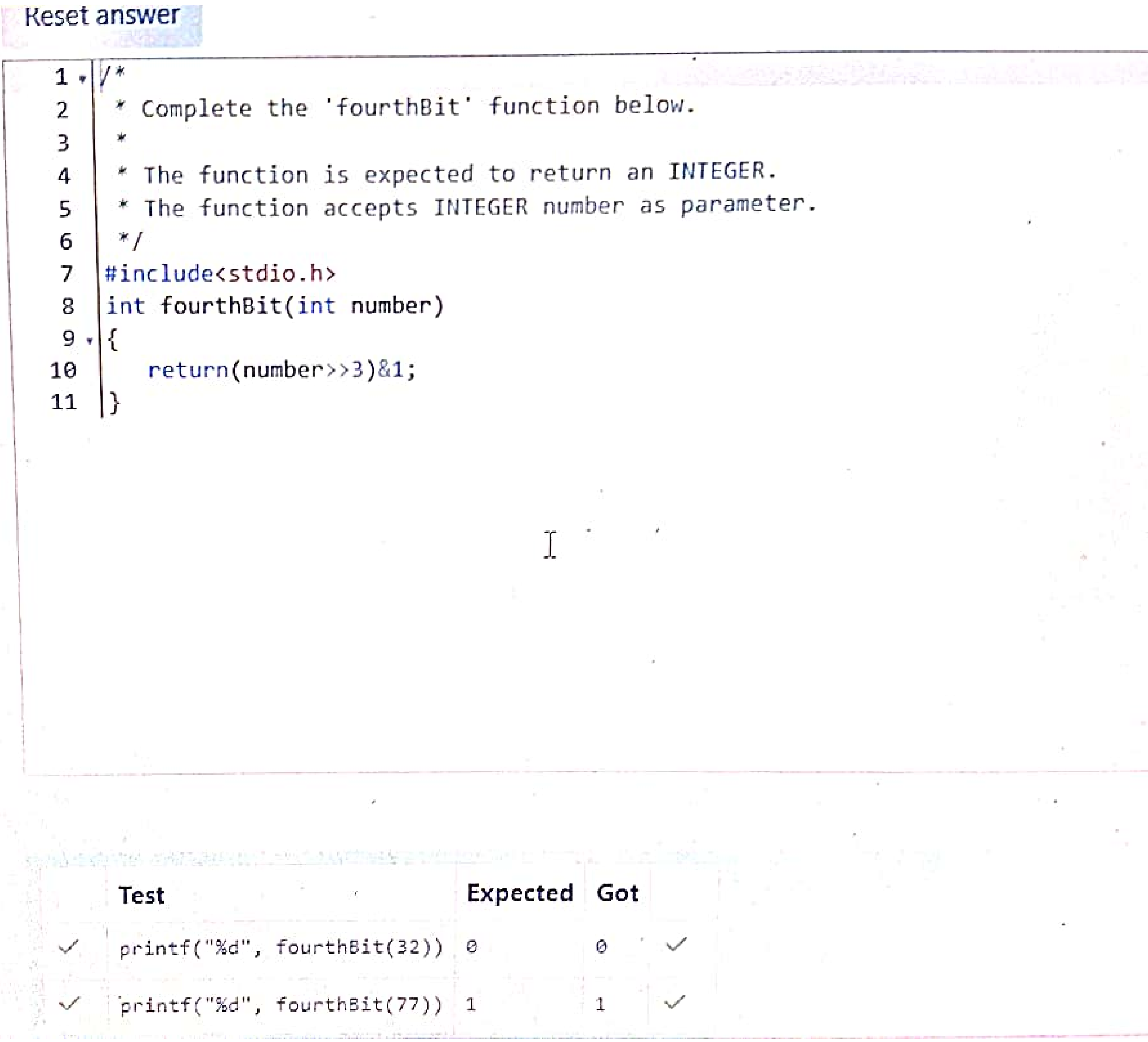
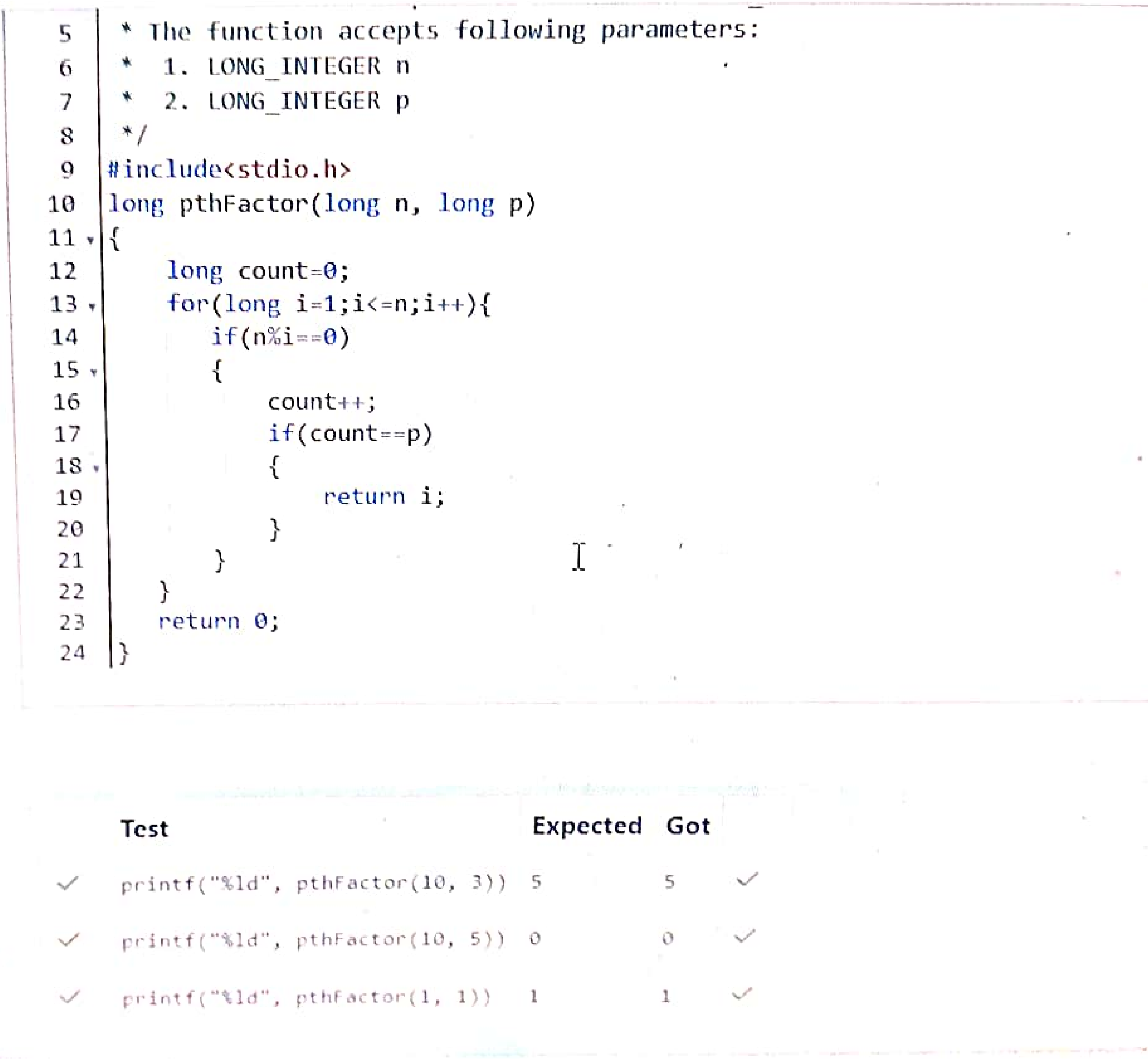


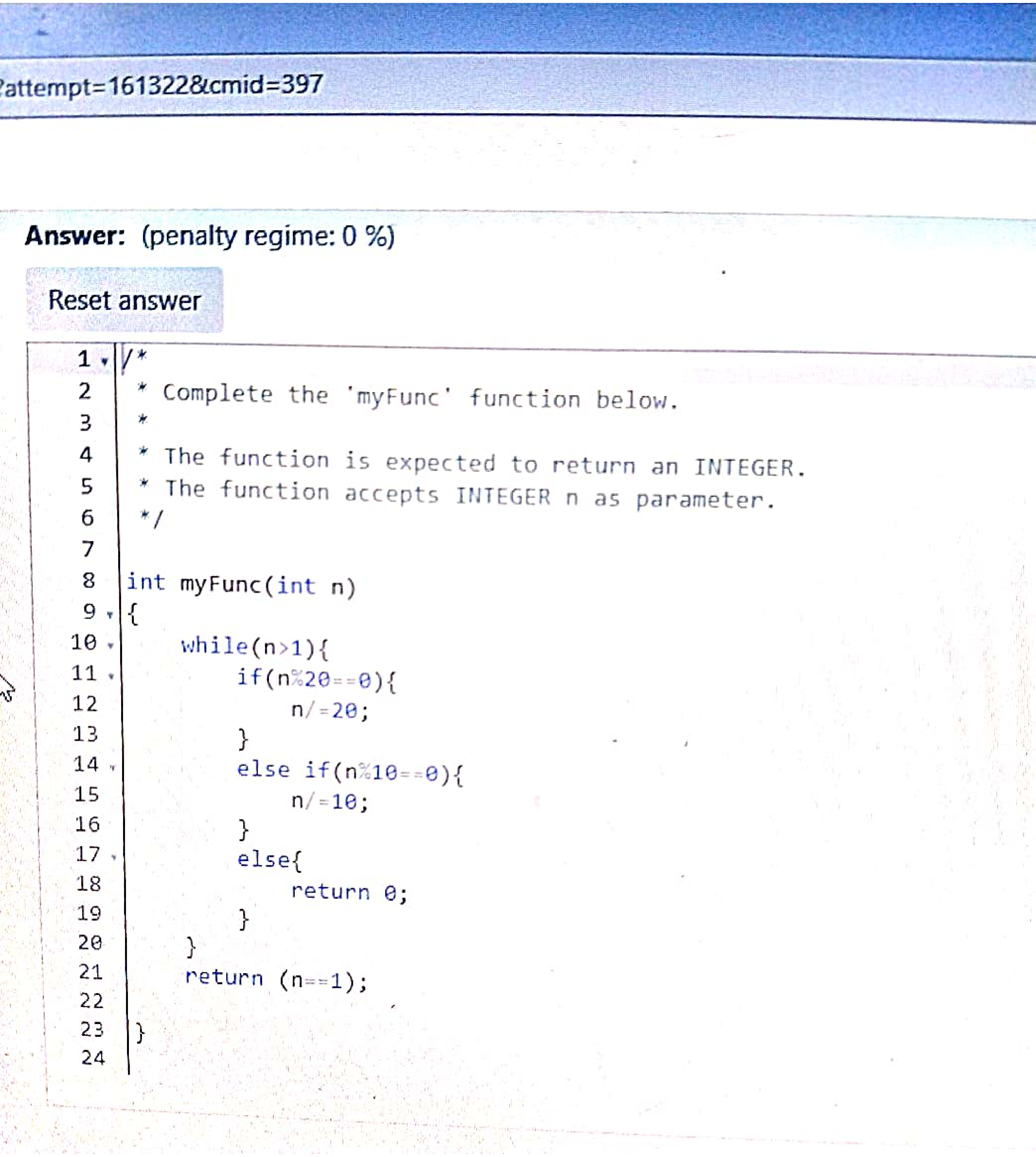
```c
/*
 * Complete the 'fourthBit' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts INTEGER number as parameter.
 */
#include<stdio.h>
int fourthBit(int number)
{
    return(number>>3)&1;
}
```

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



```
 * The function accepts following parameters:
 *  1. LONG_INTEGER n
 *  2. LONG_INTEGER p
 */
#include<stdio.h>
long pthFactor(long n, long p)
{
    long count=0;
    for(long i=1;i<=n;i++){
       if(n%i==0)
       {
           count++;
           if(count==p)
           {
               return i;
           }
       }
    }
    return 0;
}
```

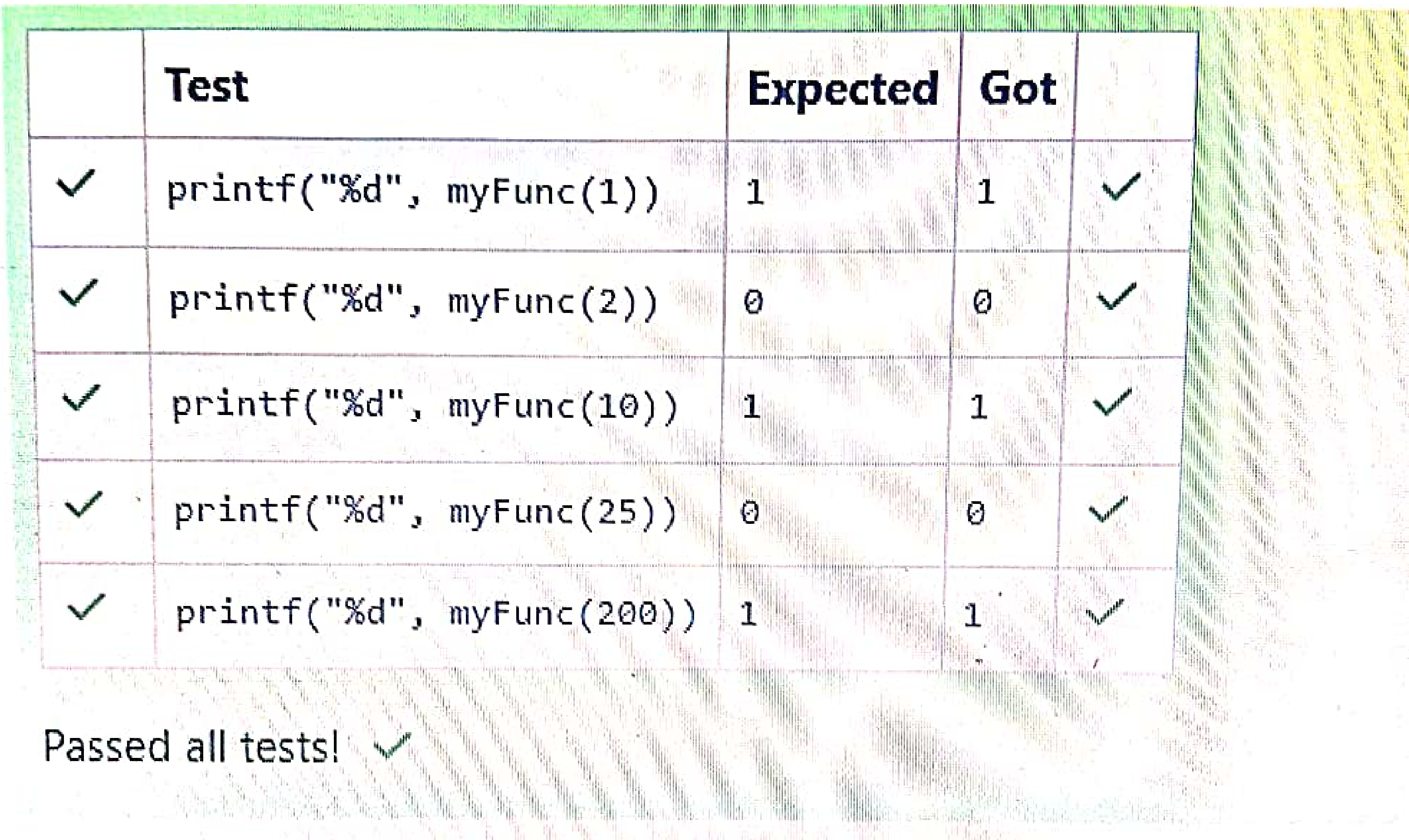
	Test	Expected	Got	
✓	printf("%ld", pthFactor(10, 3))	5	5	✓
✓	printf("%ld", pthFactor(10, 5))	0	0	✓
✓	printf("%ld", pthFactor(1, 1))	1	1	✓



?attempt=161322&cmid=397

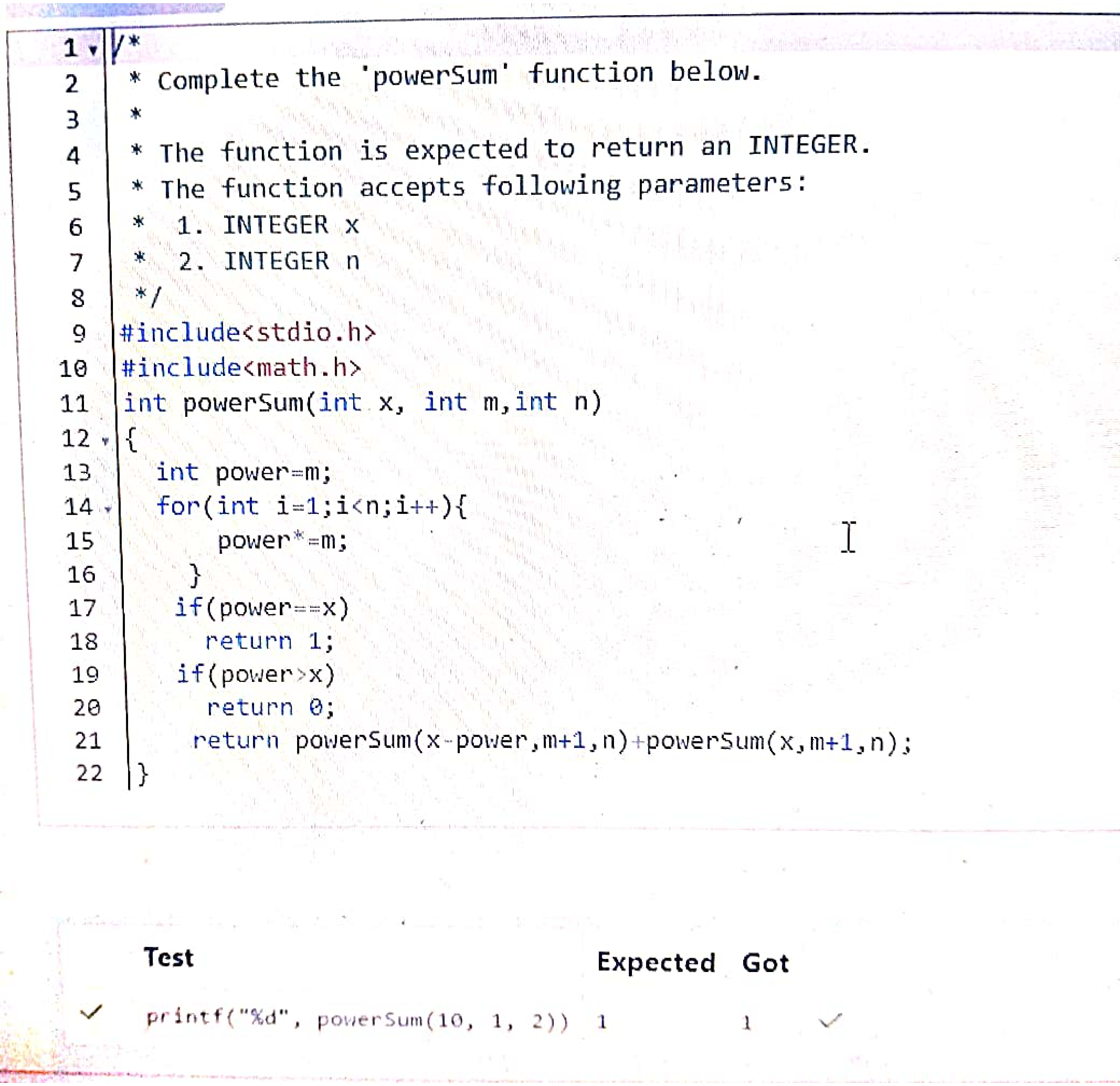
Answer: (penalty regime: 0 %)

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

int myFunc(int n)
{
    while(n>1){
        if(n%20==0){
            n/=20;
        }
        else if(n%10==0){
            n/=10;
        }
        else{
            return 0;
        }
    }
    return (n==1);

}

```



	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! ✓



```c
/*
 * Complete the 'powerSum' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts following parameters:
 *  1. INTEGER x
 *  2. INTEGER n
 */
#include<stdio.h>
#include<math.h>
int powerSum(int x, int m,int n)
{
  int power=m;
  for(int i=1;i<n;i++){
      power*=m;
    }
   if(power==x)
     return 1;
   if(power>x)
     return 0;
    return powerSum(x-power,m+1,n)+powerSum(x,m+1,n);
}
```

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