

```

1  /*
2  * Complete the 'fourthBit' function below.
3  *
4  * The function is expected to return an INTEGER.
5  * The function accepts INTEGER number as parameter.
6  */
7
8  int fourthBit(int number)
9  {
10     int binary[32];
11     int i=0;
12     while(number>0)
13     {
14         binary[i]=number%2;
15         number/=2;
16         i++;
17     }
18     if(i>=4)
19     {
20         return binary[3];
21     }
22     else
23     return 0;
24 }
25

```

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

Passed all tests! ✓

```

1  /*
2  * Complete the 'pthFactor' function below.
3  *
4  * The function is expected to return a LONG_INTEGER.
5  * The function accepts following parameters:
6  * 1. LONG_INTEGER n
7  * 2. LONG_INTEGER p
8  */
9
10 long pthFactor(long n, long p)
11 {
12     int count=0;
13     for(long i=1;i<=n;i++)
14     {
15         if(n%i==0)
16         {
17             count++;
18             if(count==p)
19             {
20                 return i;
21             }
22         }
23     }
24     return 0;
25 }

```

	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! ✓

```

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     {
12         if(n%20==0)
13         {
14             n/=20;
15         }
16         else if(n%10==0)
17         {
18             n/=10;
19         }
20         else
21         {
22             return 0;
23         }
24     }
25     return 1;
26 }
27

```

	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! ✓

```

1  /*
2  * Complete the 'powerSum' function below.
3  *
4  * The function is expected to return an INTEGER.
5  * The function accepts following parameters:
6  * 1. INTEGER x
7  * 2. INTEGER n
8  */
9
10 int powerSum(int x, int m, int n)
11 {
12     if(x==0)
13     {
14         return 1;
15     }
16     if(x<0)
17     {
18         return 0;
19     }
20     int count=0;
21     for(int i=m;i<=n;i++)
22     {
23         int power=1;
24         for(int j=0;j<n;j++)
25         {
26             power*=i;
27         }
28         if(power>x)
29         {
30             break;
31         }
32         count+=powerSum(x-power,i+1,n);
33     }
34     return count;
35 }
36

```

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

Passed all tests! ✓