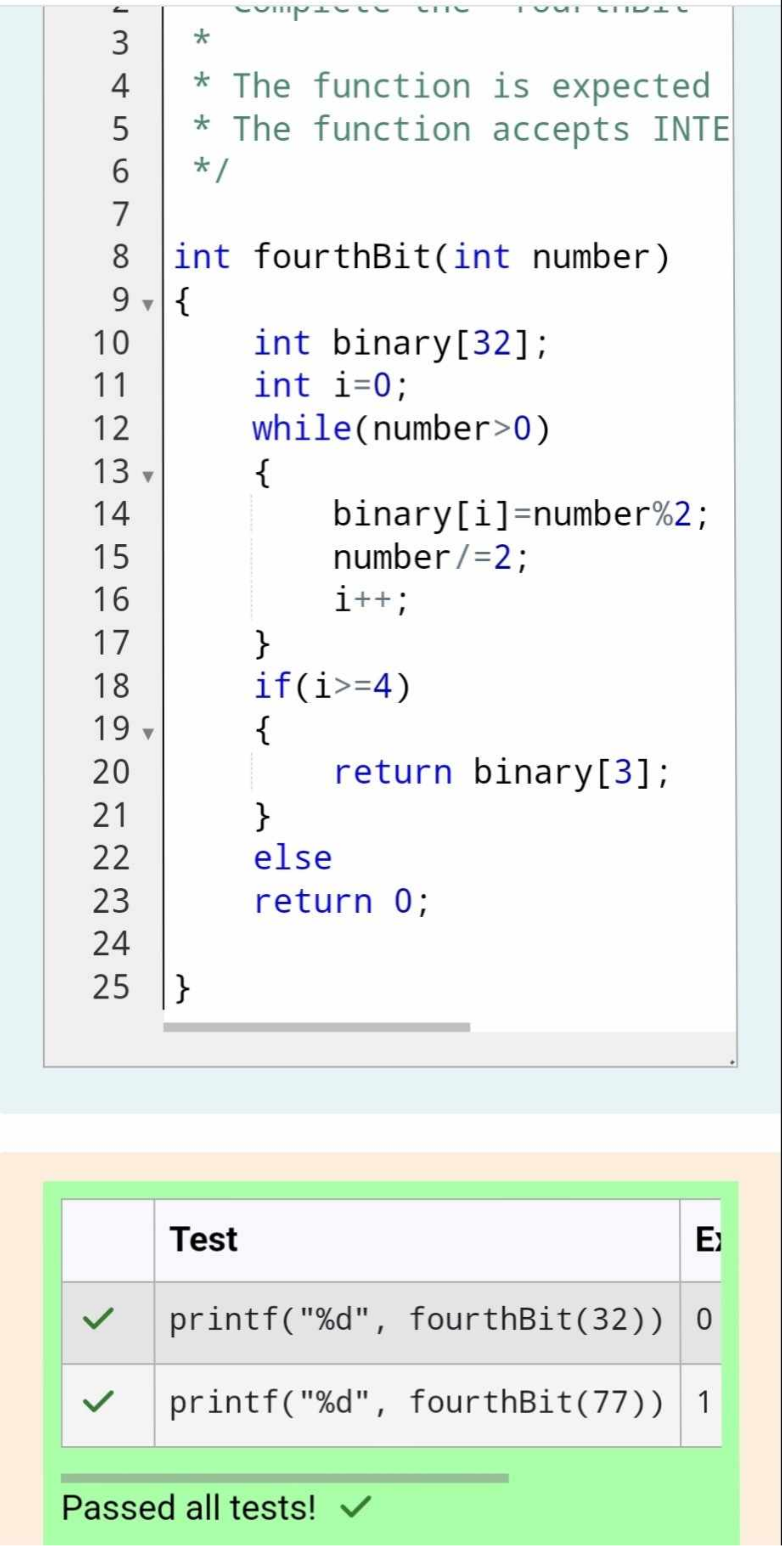
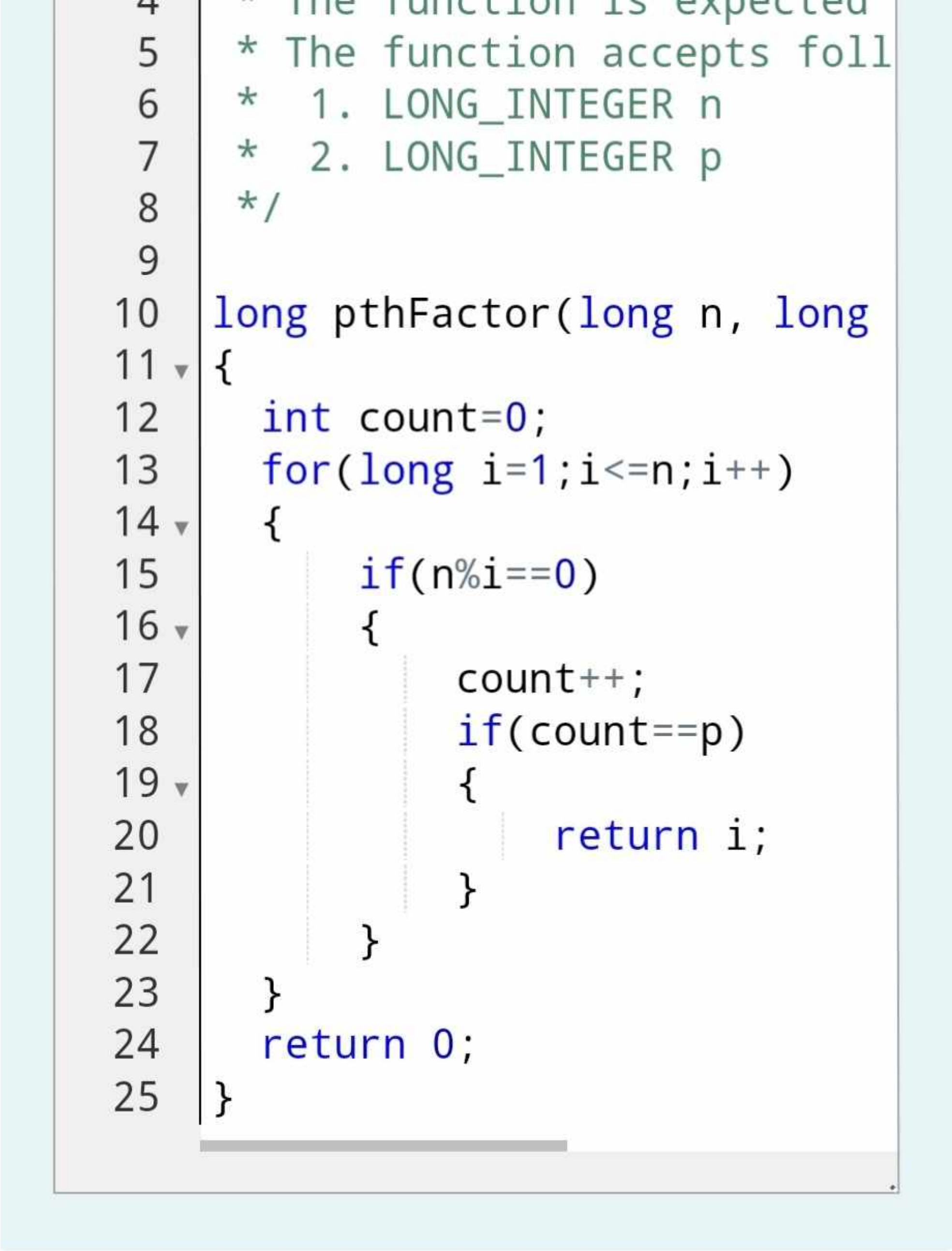


```
 * Complete the 'fourthBit
 *
 * The function is expected
 * The function accepts INTE
 */

int fourthBit(int number)
{
    int binary[32];
    int i=0;
    while(number>0)
    {
        binary[i]=number%2;
        number/=2;
        i++;
    }
    if(i>=4)
    {
        return binary[3];
    }
    else
    return 0;

}
```

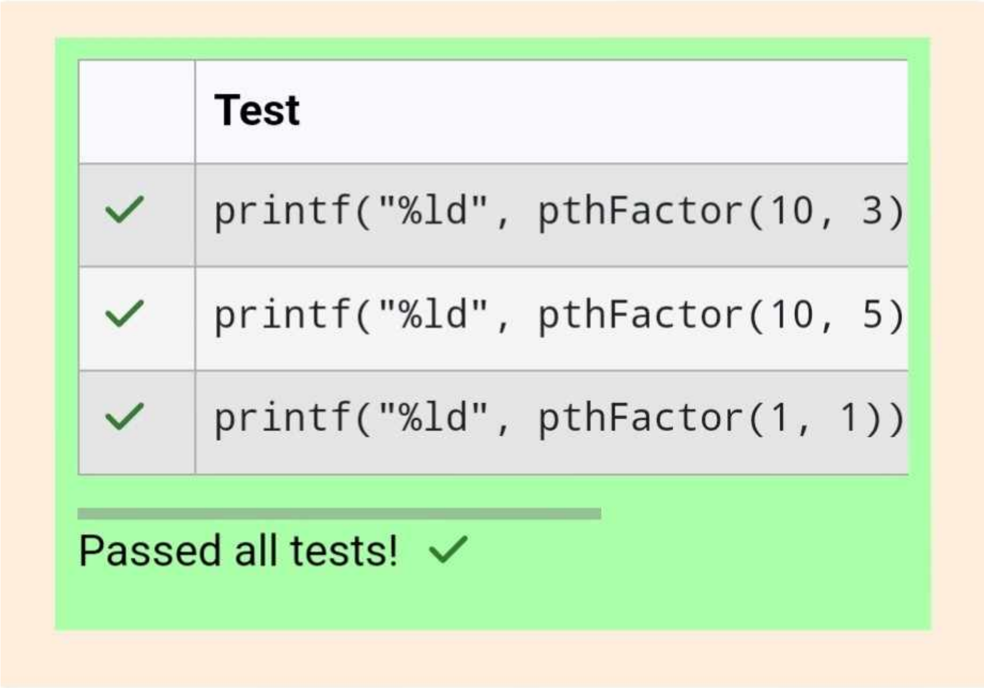
	Test	E
✓	`printf("%d", fourthBit(32))`	0
✓	`printf("%d", fourthBit(77))`	1

Passed all tests! ✓

SCAN AnyScanner



```
 * The function is expected
 * The function accepts foll
 *  1. LONG_INTEGER n
 *  2. LONG_INTEGER p
 */

long pthFactor(long n, long
{
  int count=0;
  for(long i=1;i<=n;i++)
  {
      if(n%i==0)
      {
          count++;
          if(count==p)
          {
              return i;
          }
      }
  }
  return 0;
}
```



	Test
✓	`printf("%ld", pthFactor(10, 3)`
✓	`printf("%ld", pthFactor(10, 5)`
✓	`printf("%ld", pthFactor(1, 1))`

Passed all tests! ✓

SCAN AnyScanner