Исходный вариант

```
from random import randint

n_comp = randint(1, 100)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')
n = 0 #Счетчик числа попыток
while True: #Повторение попыток
    n = n + 1 #Номер очередной попытки
    otvet = int(input('Наберите это число '))
    if otvet > n_comp:
        print('Загаданное число меньше.')
    elif otvet < n_comp:
        print('Загаданное число больше.')
    else:
        print('Правильно! Число попыток отгадывания -', n)
        break #Выход из цикла
```

Нулевой вариант

```
from random import randint
n_comp = randint(1, 100)
print(n_comp)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')

def check_numder(numder, n_comp):
    if numder > n_comp:
        return 'Загаданное число меньше'
    elif otvet < n_comp:
        return 'Загаданное число больше.'
    else:
        return f'Правильно! Число попыток отгадывания - {n}'

n = 1               # Счетчик числа попыток
while True:
    otvet = int(input('Введите Ваше число '))
    answer = check_numder(otvet, n_comp)
    print(answer)
    n = n + 1
    if otvet == n_comp:
        break

'''Замечания по реализации:
главный цикл организует логику повторения попыток и считает их количество,
вопрос - должен ли он считать или это обязанность самой функции?'"
```

Первый вариант – с глобальной переменной

```
from random import randint
n = 0               # Счетчик числа попыток

n_comp = randint(1, 100)
print(n_comp)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')

def check_numder(numder, n_comp):
    global n
    n+=1
    if numder > n_comp:
        return 'Загаданное число меньше'
    elif otvet < n_comp:
        return 'Загаданное число больше.'
    else:
        return f'Правильно! Число попыток отгадывания - {n}'
```

```
while True:
    otvet = int(input('Введите Ваше число '))
    answer = check_numder(otvet, n_comp)
    print(answer)
    if otvet == n_comp:
        break


'''Замечания по реализации:
главный цикл организует логику повторения попыток,
а функция считает их количество, ''
```

Второй вариант с атрибутом – статической переменной'

```
from random import randint
#n = 0               # Счетчик числа попыток

n_comp = randint(1, 100)
print(n_comp)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')

def check_numder(numder, n_comp):
    check_numder.n += 1
    if numder > n_comp:
        return 'Загаданное число меньше'
    elif otvet < n_comp:
        return 'Загаданное число больше.'
    else:
        return f'Правильно! Число попыток отгадывания - {check_numder.n}'
check_numder.n = 0

while True:
    otvet = int(input('Введите Ваше число '))
    answer = check_numder(otvet, n_comp)
    print(answer)
    if otvet == n_comp:
        break


'''Замечания по реализации:
главный цикл организует логику повторения попыток,
а функция считает их количество, добавляя себе атрибут и
использует как статическую переменную'''
```

Третий вариант с атрибутом с помощью специальных проверок на наличие атрибута

```
from random import randint
#n = 0               # Счетчик числа попыток

n_comp = randint(1, 100)
print(n_comp)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')

def check_numder(numder, n_comp):
    #if not hasattr(check_numder, "n"):
    #   check_numder.n = 0
    #check_numder.n += 1
    check_numder.n = getattr(check_numder, 'n', 0) + 1
    if numder > n_comp:
        return 'Загаданное число меньше'
    elif otvet < n_comp:
        return 'Загаданное число больше.'
    else:
```

```
        return f'Правильно! Число попыток отгадывания - {check_numder.n}'


while True:
    otvet = int(input('Введите Ваше число '))
    answer = check_numder(otvet, n_comp)
    print(answer)
    if otvet == n_comp:
        break
```

Четвертый вариант – с помощью декорирования

```
from random import randint
#n = 0               # Счетчик числа попыток

n_comp = randint(1, 100)
print(n_comp)
print('Компьютер "загадал" число в интервале от 1 до 100. Какое?')

def static_vars(**kwargs):
    '''декоратор для добавления атрибута для функции'''
    def decorate(func):
        for k in kwargs:
            setattr(func, k, kwargs[k])
        return func
    return decorate

@static_vars(n=0)
def check_numder(numder, n_comp):
    check_numder.n += 1
    if numder > n_comp:
        return 'Загаданное число меньше'
    elif otvet < n_comp:
        return 'Загаданное число больше.'
    else:
        return f'Правильно! Число попыток отгадывания - {check_numder.n}'


while True:
    otvet = int(input('Введите Ваше число '))
    answer = check_numder(otvet, n_comp)
    print(answer)
    if otvet == n_comp:
        break


'''
Замечания по реализации:
главный цикл организует логику повторения попыток,
а функция считает их количество, используя декорирование
'''
```