functions-in-python

May 11, 2024

1 Functions in Python - Learn by Solving 10 problems

1.1 10 Problems

1. Basic Function Syntax Problem: Write a function to calculate and return the square of a number.

2. Function with Multiple Parameters Problem: Create a function that takes two numbers as parameters and returns their sum.

3. Polymorphism in Functions Problem: Write a function multiply that multiplies two numbers, but can also accept and multiply strings.

4. Function Returning Multiple Values Problem: Create a function that returns both the area and circumference of a circle given its radius.

5. Default Parameter Value Problem: Write a function that greets a user. If no name is provided, it should greet with a default name.

6. Lambda Function Problem: Create a lambda function to compute the cube of a number.

7. Function with *args Problem: Write a function that takes variable number of arguments and returns their sum.

8. Function with **kwargs Problem: Create a function that accepts any number of keyword arguments and prints them in the format key: value.

9. Generator Function with yield Problem: Write a generator function that yields even numbers up to a specified limit.

10. Recursive Function Problem: Create a recursive function to calculate the factorial of a number.

1. Basic Function Syntax Problem: Write a function to calculate and return the square of a number.

[ ]:
```python
def square_of_num(number):
  return number ** 2

result=square_of_num(5)
print(result)
```

```
25
```

2. Function with Multiple Parameters Problem: Create a function that takes two numbers as parameters and returns their sum.

```
[ ]: def sum_of_two(a,b):
       return a+b

     print(sum_of_two(5,6))
```

```
11
```

3. Polymorphism in Functions Problem: Write a function multiply that multiplies two numbers, but can also accept and multiply strings.

```
[ ]: def multiply(pOne,pTwo):
       return pOne*pTwo

     print(multiply(5,5))
     print(multiply(5,"a"))
     print(multiply("a",5))
     # print(multiply("a","5")) //error - TypeError: can't multiply sequence by␣
       ↪non-int of type 'str'
```

```
25
aaaaa
aaaaa
```

4. Function Returning Multiple Values Problem: Create a function that returns both the area and circumference of a circle given its radius.

```
[ ]: import math
     def circle_calc(r):
       circumference = round(2*math.pi*r,2)
       area = round(math.pi*r*r,2)
       return area,circumference

     a,c=circle_calc(7)

     print("Area: ",a, "Circumference: ",c)
```

```
Area:  153.94 Circumference:  43.98
```

5. Default Parameter Value Problem: Write a function that greets a user. If no name is provided, it should greet with a default name.

```
[ ]: def greet(name="User"):
       return "Hello, "+name+" "

     print(greet("Gautam"))
     print(greet())
```

```
Hello, Gautam
Hello, User
```

6. Lambda Function Problem: Create a lambda function to compute the cube of a number.

[ ]:
```
cube = lambda x: x ** 3
print(cube(3))
```

```
27
```

7. Function with *args Problem: Write a function that takes variable number of arguments and returns their sum.

[ ]:
```
def sum_all(*args):
  return sum(args) # using sum function

print(sum_all(1,2))
print(sum_all(1,2,3))
print(sum_all(1,2,3,4,5))
```

```
3
6
15
```

[ ]:
```
def sum_all(*args):
  print(*args)
  print(args) #tuple - which is iterable
  sum = 0
  for i in args: # don't use *args here
    sum = sum+i
  return sum

print(sum_all(1,2))
print(sum_all(1,2,3))
print(sum_all(1,2,3,4,5))
```

```
1 2
(1, 2)
3
1 2 3
(1, 2, 3)
6
1 2 3 4 5
(1, 2, 3, 4, 5)
15
```

8. Function with **kwargs Problem: Create a function that accepts any number of keyword arguments and prints them in the format key: value.

[ ]:
```
def print_kwargs(**kwargs):
    for key, value in kwargs.items():
        print(f"{key}: {value}")
```

```python


print_kwargs(name="shaktiman", power="lazer")
print_kwargs(name="shaktiman")
print_kwargs(name="shaktiman", power="lazer", enemy = "Dr. Jackaal")
```

```
name: shaktiman
power: lazer
name: shaktiman
name: shaktiman
power: lazer
enemy: Dr. Jackaal
```

9. Generator Function with yield Problem: Write a generator function that yields even numbers up to a specified limit.

[ ]:
```python
def even_generator(limit):
    for i in range(2, limit + 1, 2):
        yield i



for num in even_generator(10):
    print(num)
```

```
2
4
6
8
10
```

10. Recursive Function Problem: Create a recursive function to calculate the factorial of a number.

[ ]:
```python
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

print(factorial(5))
```

```
120
```