1. What is a lambda function in Python, and how does it differ from a regular function?

Answer:

Lambda function are different from regular functions because lambda functions are anonymous functions that are defined without a name and parentheses. We use lambda function to evaluate short expressions.

2. Can a lambda function in Python have multiple arguments? If yes, how can you define and use

Them?

Answer:

We can have multiple arguments in lambda function.

add = lambda x, y : x + y

print(add(10, 20))

Output:

30

3. How are lambda functions typically used in Python? Provide an example use case.

Answer:

We use lambda functions as an argument to higher order functions. These are the functions that takes other functions as an argument.

squares = lambda x: x\*x

print(squares(5))

Output:

25

4. What are the advantages and limitations of lambda functions compared to regular functions in

Python?

Advantages are:

The lambda functions can be immediately passed without the need for variables.

They can have single line of code.

They return automatically.

Limitations are:

Lambda functions can only have one expression.

Lambda functions cannot have a docstring.

5. Are lambda functions in Python able to access variables defined outside of their own scope?

Explain with an example.

Answer:

Lambda functions have their own local namespace and cannot access variables other than those in parameter list and those in lobal namespace.

6. Write a lambda function to calculate the square of a given number.

Answer:

squares = lambda x: x\*x

print(squares(5))

Output:

25

7. Create a lambda function to find the maximum value in a list of integers.

Answer:

list=[1,2,3,4,5]

lambda\_max = lambda x: max(x)

lambda\_max(list)

Output:  
5

8. Implement a lambda function to filter out all the even numbers from a list of integers.

Answer:

nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

even = list(filter(lambda x: x%2 == 0, nums))

print(even)

Output:

[2,4,6,8,10]

9. Write a lambda function to sort a list of strings in ascending order based on the length of each

String.

Answer:

List = ["ab", "abcd", "abcdef", "a", "abcdefgh", "abc"]

sortedList = sorted(List, key=lambda x: len(x))

print(sortedList)

Output:

['a', 'ab', 'abc', 'abcd', 'abcdef', 'abcdefgh']

10. Write a recursive function to calculate the factorial of a given positive integer.

Answer:

def factorial(n):

if n == 0 or n == 1:

return 1

else:

return n \* factorial(n - 1)

n= 5

result = factorial(n)

print(f"The factorial of {n} is: {result}")

Output:

The factorial of 5 is: 120

11. Implement a recursive function to compute the nth Fibonacci number.

Answer:

def fibonacci(n):

if n <= 0:

raise ValueError("Input must be a positive integer.")

elif n == 1 or n == 2:

return 1

else:

return fibonacci(n - 1) + fibonacci(n - 2)

n=10

fibonacci(n)

Output:

55

12. Create a recursive function to find the sum of all the elements in a given list.

Answer:

def sum(lst):

if len(lst) == 0:

return 0

elif len(lst) == 1:

return lst[0]

else:

return lst[0] + sum(lst[1:])

lst = [1, 2, 3, 4, 5]

result = sum(lst)

print(result)

Output:

15

13. Write a recursive function to determine whether a given string is a palindrome.

Answer:

def palindrome(s):

if len(s) < 1:

return True

else:

if s[0] == s[-1]:

return palindrome(s[1:-1])

a=str(input("Enter string:"))

if(palindrome(a)==True):

print("String is a palindrome!")

else:

print("String isn't a palindrome!")

Output:

Enter string:malayalam

String is a palindrome!

14. Implement a recursive function to find the greatest common divisor (GCD) of two positive integers.

Answer:

def gcd(a, b):

if b == 0:

return a

else:

return gcd(b, a % b)

a = 26

b = 18

result = gcd(a, b)

print("The GCD of", a, "and", b, "is", result)

Output:

The GCD of 26 and 18 is 2