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

Ans: A lambda function is an anonymous function that can take any number of argument but unlike

Normal function, evaluate and returns only one expression.unlike a normal function we don’t surround the parameters of a with a parentheses.

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

Ans : yes a lambda function in python have multiple arguments.

Below is the example of lambda function using multiple argument

even\_or\_odd = lambda x: 'even'if x%2 == 0 else'odd'

print(even\_or\_odd(12))

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

Ans: Lambda functions are frequently used with higher order function. Which take one or more fuction as arguments or return one or more functions.

Add =lambda x,y : x + y

Print(add(30,10)

1. What are the advantages and limitations of lambda functions compared to regular functions in Python?

Ans : Advantages: In lambda function you no need to define the function , you can pass argument immediately no variable needed,

Limtation : in lambda function used for single line code only

and they don’t have docstring and thy don’t have name.,lambda fuction is not easy to understand .

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

Explain with an example.

Ans : lambda function have their own local namespace and cannot access variables other than those in their parameter list and those in the global namespace

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

square = lambda x : x\*\*2

print(square())

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

max\_num = lambda x,y: x if x > y else y

print(max\_num())

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

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

print(even\_nums)

.

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

Ans:

sorted \_products = sorted(prodcuts ,key = lambda x :x ['length'])

for products in sorted\_products:

print(products)

1. Create a lambda function that takes two lists as input and returns a new list containing the common elements between the two lists.

Ans: common\_number = list(filter(lambda x:x in a,b))

print("The common elements in two lists are : ",common\_number)

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

Ans :

def rec\_sum(n):

if n==1:

return 1

else:

return n + rec\_sum (n-1)

print(rec\_sum())

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

def fabonacci(n):

if n <= 1:

return n

else:

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

print(fabonacci())

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

Ans:

def getsum(num):

if len(num)==0:

return 0

else:

return num [0] + getsum(num[1:])

print(getsum(num))

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

Ans:

def ispalindrome(string):

if len(string)>1:

return True

else:

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

return ispalindrome(string[1:-1])

string =input("Enter the string: ")

if(ispalindrome(string)==True):

print("The stirng is a palindrome")

else:

print("The string is not palindrome")

print(ispalindrome(string))

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

Ans:

def gcd(a,b):

if(b==0):

return a

else:

return gcd(b,a%b)

a = int(input("Enter the first number: "))

b = int(input("Enter the second number: "))

GCD = gcd(a,b)

print("GCD is: ")

print(GCD)