24BIT167 - VANDITA NAWANI

AIM : To implement various operations on functions in Python, including calling and defining of a function

System Configration : Operating system: Windows 11, Google Colab

Theory : Function is a block of reusable code that performs specific task. Function helps the code to reuse, modularity, and better organization

Write a program that defines a function count_lower_upper() that accepts a string and calculates the number of uppercase and lowercase alphabets in it. It should return these values as a dictionary. Call this function for some sample string.

```
def count_lower_upper(string):
    lower_count = 0
    upper_count = 0

    for char in string:
        if char.islower():
            lower_count += 1
        elif char.isupper():
            upper_count += 1

    result = {"lower_count": lower_count, "upper_count": upper_count}
    return result


string = input("ENTER A STRING:")
count_lower_upper(string)
```

```
ENTER A STRING:VanDIta
{'lower_count': 4, 'upper_count': 3}
```

Write a program that defines a function compute() that calculates the value of n + nn + nnn + nnnn, where n is digit received by the function. test the function for digits 4 to 7.

```
def compute(n):
    nn = int(str(n) * 2)
    nnn = int(str(n) * 3)
    nnnn = int(str(n) * 4)
    result = n + nn + nnn + nnnn
    return result

n = int(input("ENTER THE NUMBER:"))
compute(n)
```

```
ENTER THE NUMBER:4
4936
```

Write a program that defines a function create_array() to create and return a 3D array whose dimensions are passed to the function. Also initialize each element of this aray to a value passed to the function. e.g. create_array(3,4,5,n) where first three arguments are 3D array dimensions and 4th value is for initialing each value of the 3D array.

```
def create_array(a,b,c,d):
  return [[[d for k in range (c)] for j in range(b)] for i in range (a)]

arr = create_array(3,4,1,2)
print(arr)
```

```
[[[2], [2], [2], [2]], [[2], [2], [2], [2]], [[2], [2], [2], [2]]]
```

If a positive integer is entered through the keyword, write a recursive function to obtain the prime factors of the number.

```
def prime_factors(n, i=2):
  if n==1:
    return []
  if n%i==0:
    return[i] + prime_factors(n//i, i)
  else:
    return prime_factors(n, i+1)
```

```
num = int(input("ENTER THE NUMBER:"))

if num >0:
  fac = prime_factors(num)
  print("THE FACTORS OF THE NUMBERS ARE:", fac)
else:
  print("PLEASE ENTER A VALID NUMBER")
```

```
ENTER THE NUMBER:88
THE FACTORS OF THE NUMBERS ARE: [2, 2, 2, 11]
```

A positive integer is entered through the keyboard. Write a function to find its binary equivalent of this number.

```
def to_binary(n):
  binary = ""
  while n>0:
    binary = str(n%2) + binary
    n= n//2
  return binary


num = int(input("ENTER A NUMBER:"))

if num >0:
  print("THE BINARY EQUIVALENT IS :", to_binary(num))
else:
  print("PLEASE ENTER A VALID NUMBER")
```

```
ENTER A NUMBER:4
THE BINARY EQUIVALENT IS : 100
```

A string is entered through the keyboard. Write a recursive function that counts the number of vowels in this string.

```
def count_vowel(n):
  count = 0
  for char in n:
    if char.lower() in "aeiou":
      count+=1
  return count


str = input("ENTER A STRING:")
count_vowel(str)
```

```
ENTER A STRING:vandita
3
```

Write a recursive function that reverses the list of numbers that it receives.

```
 def reverse_list(n):
  if not n:
    return []
  else:
    return [n[-1]] + reverse_list(n[:-1])


list = [1,2,3,4,5]
reverse_list(list)
```

```
[5, 4, 3, 2, 1]
```

Calculate a^b where a and b received through the keyword using recursion.

```
def pow(n,m):
  if m==0:
    return 1
  else:
    return n**m
```

```
num1 = int(input("ENTER THE BASE OF EXPONENT:"))
num2 = int(input("ENTER THE POWER OF THE EXPONENT:"))
pow(num1, num2)
```

```
ENTER THE BASE OF EXPONENT:5
ENTER THE POWER OF THE EXPONENT:4
625
```

Write a recursive function to obtain average of all numbers present in a given list.

```
def avg(lst):
  if not lst:
    return 0
  if len(lst)==1:
    return lst[0]
  return (lst[0] + (len(lst)-1)*avg(lst[1:])/len(lst))


num = [1,2,3,5,6,7]
print("THE AVERAGE OF THE NUMBERS IN LIST IS :", avg(num))
```

```
THE AVERAGE OF THE NUMBERS IN LIST IS : 10.333333333333334
```