Darshan UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2301CS404

Lab - 8

Jeet Bhalodi (23031701006)
28-01-2024

User Defined Function

01) Write a function to calculate BMI given mass and height. (BMI = mass/h**2)

In [7]:
```
def calculate_bmi(mass, height):
    bmi = mass / (height ** 2)
    return bmi

mass = int(input("Enter Mass : "))
height = float(input("Enter Height : "))
print(f"The BMI is {calculate_bmi(mass, height): .2f}")
```

```
The BMI is  18.52
```

02) Write a function that add first n numbers.

In [9]:
```
def sum_of_first_n_numbers(n):
    total_sum = 0
    for i in range(1, n + 1):
        total_sum += i
    return total_sum

n=int(input("Enter N : "))
print(f"The sum of the first {n} numbers is : {sum_of_first_n_numbers(n)}")
```

```
The sum of the first 5 numbers is : 15
```

03) Write a function that returns 1 if the given number is Prime or 0 otherwise.

In [16]:
```
def is_prime(n):
    if n <= 1:
        return 0
    for i in range(2, n):
        if n % i == 0:
            return 0
    return 1

n=int(input("Enter Number : "))
print(f"Is {n} a prime number? : {is_prime(n)}")
```

```
Is 5 a prime number? : 1
```

04) Write a function that returns the list of Prime numbers between given two numbers.

In [14]:
```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True

def prime_numbers_between(start, end):
    primes = []
    for number in range(start, end + 1):
        if is_prime(number):
            primes.append(number)
    return primes


start = int(input("Enter start : "))
end = int(input("Enter end : "))
print(f"Prime numbers between {start} and {end}: {prime_numbers_between(start, end)
```

```
Prime numbers between 1 and 11: [2, 3, 5, 7, 11]
```

05) Write a function that returns True if the given string is Palindrome or False otherwise.

In [23]:
```
def is_palindrome(s):

    str1 = ''.join(char.lower() for char in s if char.isalnum())

    return str1 == str1[::-1]
```

```
string = input("Enter String : ")
print(f"Is the string \"{string}\" a palindrome? {is_palindrome(string)}")
```

```
Is the string "abcba" a palindrome? True
```

06) Write a function that returns the sum of all the elements of the list.

In [27]:
```
def sum_of_elements(lst):
    total_sum = 0
    for element in lst:
        total_sum += element
    return total_sum

element = input("Enter Number With Comma : ")
list1=[int(i) for i in element.split(',')]
print(f"The sum of all elements in the list is : {sum_of_elements(list1)}")
```

```
The sum of all elements in the list is : 25
```

07) Write a function to calculate the sum of the first element of each tuples inside the list.

In [31]:
```
def sum_of_first_elements():
    user_input = input("Enter a list of tuples (EX.(1, 2)): ")

    list_of_tuples = eval(user_input)

    total_sum = sum(i[0] for i in list_of_tuples)

    return total_sum

print(f"The sum of the first elements in the list of tuples is : {sum_of_first_elem
```

```
The sum of the first elements in the list of tuples is : 9
```

08) Write a recursive function to find nth term of Fibonacci Series.

In [33]:
```
def fibonacci(n):
    if n <= 0:
        return "Input should be a positive integer."
    elif n == 1:
        return 0
    elif n == 2:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)


n=int(input("Enter Number : "))
print(f"The {n}th term of the Fibonacci series is : {fibonacci(n)}")
```

```
The 10th term of the Fibonacci series is : 34
```

09) Write a function to get the name of the student based on the given rollno.

Example: Given dict1 = {101:'Ajay', 102:'Rahul', 103:'Jay', 104:'Pooja'} find name of student whose rollno = 103

In [35]:
```
def get_student_name(student_dict,rollno):

    return student_dict.get(rollno, "Roll number not found")

student_dict = {}
num_entries = int(input("How many students do you want to enter? "))

for _ in range(num_entries):
    rollno = int(input("Enter roll number: "))
    name = input("Enter name: ")
    student_dict[rollno] = name

rollno = int(input("Enter the roll number to search for: "))
print(f"The name of the student is {get_student_name(student_dict,rollno)}")
```

```
The name of the student is jeet
```

10) Write a function to get the sum of the scores ending with zero.

Example : scores = [200, 456, 300, 100, 234, 678]

Ans = 200 + 300 + 100 = 600

In [37]:
```
def sum_of_scores_ending_with_zero(scores):
    total_sum = 0
    for i in scores:
        if i % 10 == 0:
            total_sum += i
    return total_sum

element = input("Enter Number With Comma : ")
scores=[int(i) for i in element.split(',')]
print(f"The sum of the scores ending with zero is : {sum_of_scores_ending_with_zero
```

```
The sum of the scores ending with zero is : 1140
```

11) Write a function to invert a given Dictionary.

hint: keys to values & values to keys

Before : {'a': 10, 'b':20, 'c':30, 'd':40}

After : {10:'a', 20:'b', 30:'c', 40:'d'}

In [43]:
```
def invert_dictionary(d):
    return {value: key for key, value in d.items()}

dict1 = {}
entries = int(input("How many entries do you want to enter? "))

for _ in range(entries):
    key = int(input("Enter key : "))
    values = input("Enter values : ")
    dict1[key] = values
dict2 = invert_dictionary(dict1)
print(f"Original dictionary: {dict1}")
print(f"Inverted dictionary: {dict2}")
```

```
Original dictionary: {1: 'a', 2: 'b', 3: 'c'}
Inverted dictionary: {'a': 1, 'b': 2, 'c': 3}
```

12) Write a function to check whether the given string is Pangram or not.

hint: Pangram is a string containing all the characters a-z atlest once.

"the quick brown fox jumps over the lazy dog" is a Pangram string.

In [45]:
```
def is_pangram(s):
    alphabet = set('abcdefghijklmnopqrstuvwxyz')

    s_set = set(s.lower())

    return alphabet.issubset(s_set)

string = "the quick brown fox jumps over the lazy dog"
print(f"Is the string \"{string}\" a pangram? : {is_pangram(string)}")
```

```
Is the string "the quick brown fox jumps over the lazy dog" a pangram? : True
```

13) Write a function that returns the number of uppercase and lowercase letters in the given string.

example : Input : s1 = AbcDEfgh ,Ouptput : no_upper = 3, no_lower = 5

In [47]:
```
def count_upper_lower(s):
    no_upper = sum(1 for char in s if char.isupper())
    no_lower = sum(1 for char in s if char.islower())
    return no_upper, no_lower

s1 = input("Enter String")
no_upper, no_lower = count_upper_lower(s1)
print(f"Input: {s1}\nOutput: no_upper = {no_upper}, no_lower = {no_lower}")
```

```
Input: JeEt BhAlOdI
Output: no_upper = 6, no_lower = 5
```

14) Write a lambda function to get smallest number from the given two numbers.

In [51]:
```
min_number = lambda a, b: a if a < b else b

a = int(input("Enter number1 : "))
b = int(input("Enter number2 : "))
print(f"The smallest number is : {min_number(a, b)}")
```

```
The smallest number is : 9
```

15) For the given list of names of students, extract the names having more that 7 characters. Use filter().

In [53]:
```
def seven_chars(name):
    return len(name) > 7

student_names = ["Alexander", "Ben", "Catherine", "Daniel", "Emily", "Francesca", "

long_names = list(filter(seven_chars, student_names))

print(f"Names with more than 7 characters: {long_names}")
```

```
Names with more than 7 characters: ['Alexander', 'Catherine', 'Francesca']
```

16) For the given list of names of students, convert the first letter of all the names into uppercase. use map().

In [55]:
```
def name_first_upper(name):
    return name.capitalize()

student_names = ['jeet','bhavy','bhalodi']

ans = list(map(name_first_upper, student_names))

print(f"Names with more than 7 characters: {ans}")
```

```
Names with more than 7 characters: ['Jeet', 'Bhavy', 'Bhalodi']
```

17) Write udfs to call the functions with following types of arguments:

1. Positional Arguments
2. Keyword Arguments
3. Default Arguments
4. Variable Legngth Positional(*args) & variable length Keyword Arguments (*kwargs)
5. Keyword-Only & Positional Only Arguments

In [59]:
```
# Positional Arguments
def positional_args(a, b):
    return a + b
```

```
print("Positional Arguments : ",positional_args(3, 5))

# Keyword Arguments
def keyword_args(a, b):
    return a * b
print("Keyword Arguments : ",keyword_args(a=3, b=5))

# Default Arguments
def default_args(a, b=5):
    return a - b
print("Default Arguments : ",default_args(10))
print("Default Arguments : ",default_args(10, 3))

# Variable Legngth Positional(*args)
def var_length_positional_args(*args):
    return sum(args)
print("Variable Legngth Positional(*args) : ",var_length_positional_args(1, 2, 3, 4

# variable length Keyword Arguments (**kwargs)
def var_length_keyword_args(**kwargs):
    return ", ".join(f"{key}={value}" for key, value in kwargs.items())
print("variable length Keyword Arguments (**kwargs) : ",var_length_keyword_args(a=1

# Keyword-Only
def keyword_only_args(*, a, b):
    return a / b
print("Keyword-Only : ",keyword_only_args(a=10, b=2))

# Positional Only
def positional_only_args(a, b, /):
    return a ** b
print("Positional Only : ",positional_only_args(2, 3))
```

```
Positional Arguments :  8
Keyword Arguments :  15
Default Arguments :  5
Default Arguments :  7
Variable Legngth Positional(*args) :  10
variable length Keyword Arguments (**kwargs) :  a=1, b=2, c=3
Keyword-Only :  5.0
Positional Only :  8
```

In [ ]: