Experiment 7 - Demonstration of Functions, Lambda, and Recursion

Name: Mohammad Sayeed Kazi

Roll no : C56   Div: C Class : TY CSE

1. Largest of three numbers

Section: User-defined functions

Problem Statement:

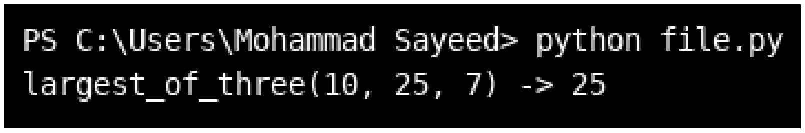
Write a user-defined function to find the largest of three numbers.

Code:

```
def largest_of_three(a, b, c):
    return max(a, b, c)

print("largest_of_three(10, 25, 7) ->", 
largest_of_three(10,25,7))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
largest_of_three(10, 25, 7) -> 25
```

2. Factorial

Section: User-defined functions

Problem Statement:

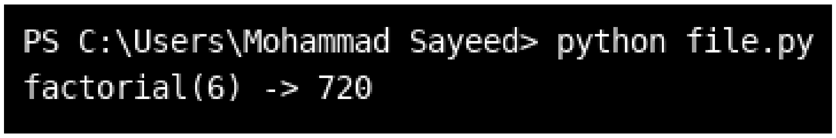
Create a function to calculate the factorial of a given number.

Code:

```
def factorial(n):
    if n < 0:
        raise ValueError("Negative not allowed")
    res = 1
    for i in range(2, n+1):
        res *= i
    return res

print("factorial(6) ->", factorial(6))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
factorial(6) -> 720
```

3. Prime check

Section: User-defined functions

Problem Statement:

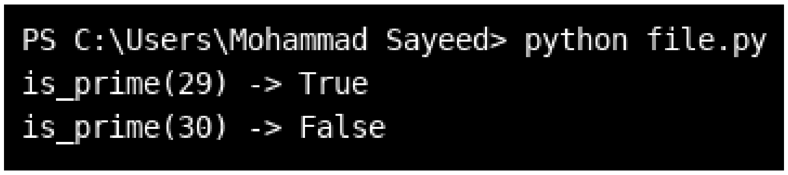
Define a function to check whether a given number is prime or not.

Code:

```
def is_prime(n):
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0:
        return False
    i = 3
    while i*i <= n:
        if n % i == 0:
            return False
        i += 2
    return True

print("is_prime(29) ->", is_prime(29))
print("is_prime(30) ->", is_prime(30))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
is_prime(29) -> True
is_prime(30) -> False
```

4. Count vowels

Section: User-defined functions

Problem Statement:

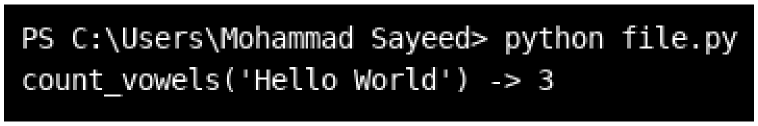
Write a function that takes a string as input and returns the number of vowels.

Code:

```
def count_vowels(s):
    return sum(1 for ch in s.lower() if ch in 'aeiou')

print("count_vowels('Hello World') ->", count_vowels("Hello
World"))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
count_vowels('Hello World') -> 3
```

5. Celsius to Fahrenheit

Section: User-defined functions

Problem Statement:

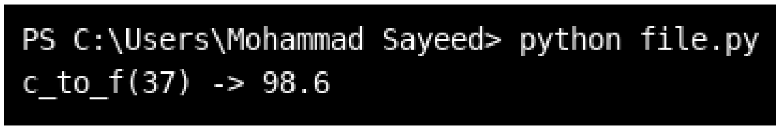
Create a function that converts temperature from Celsius to Fahrenheit.

Code:

```
def c_to_f(c):
    return (c * 9/5) + 32

print("c_to_f(37) ->", c_to_f(37))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
c_to_f(37) -> 98.6
```

6. Sum of even numbers in list

Section: User-defined functions

Problem Statement:

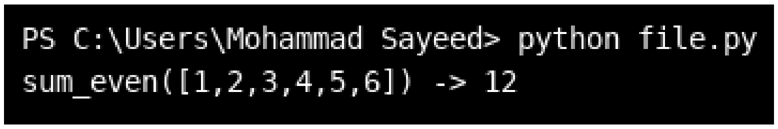
Define a function to find the sum of all even numbers in a list.

Code:

```
def sum_even(lst):
    return sum(x for x in lst if x % 2 == 0)

print("sum_even([1,2,3,4,5,6]) ->", sum_even([1,2,3,4,5,6]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
sum_even([1,2,3,4,5,6]) -> 12
```

7. Reverse string

Section: User-defined functions

Problem Statement:

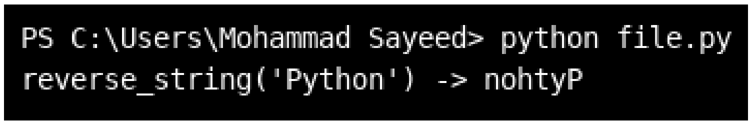
Write a function that reverses a given string.

Code:

```
def reverse_string(s):
    return s[::-1]

print("reverse_string('Python') ->", reverse_string("Python"))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
reverse_string('Python') -> nohtyP
```

8. Palindrome string check

Section: User-defined functions

Problem Statement:

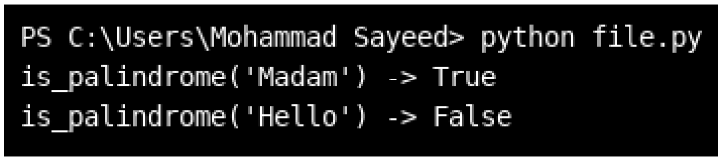
Create a function that checks whether a string is a palindrome.

Code:

```
def is_palindrome(s):
    s2 = ''.join(ch.lower() for ch in s if ch.isalnum())
    return s2 == s2[::-1]

print("is_palindrome('Madam') ->", is_palindrome("Madam"))
print("is_palindrome('Hello') ->", is_palindrome("Hello"))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
is_palindrome('Madam') -> True
is_palindrome('Hello') -> False
```

9. Max and Min from list

Section: User-defined functions

Problem Statement:

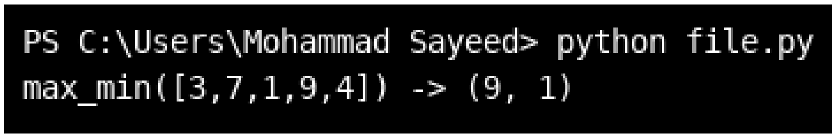
Define a function that returns the maximum and minimum values from a list.

Code:

```
def max_min(lst):
    return max(lst), min(lst)

print("max_min([3,7,1,9,4]) ->", max_min([3,7,1,9,4]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
max_min([3,7,1,9,4]) -> (9, 1)
```

10. Count words in sentence

Section: User-defined functions

Problem Statement:

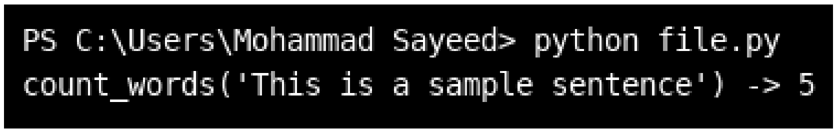
Write a function to count the number of words in a sentence.

Code:

```
def count_words(s):
    return len(s.split())

print("count_words('This is a sample sentence') ->",
count_words("This is a sample sentence"))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
count_words('This is a sample sentence') -> 5
```

11. Percentage from marks

Section: User-defined functions

Problem Statement:

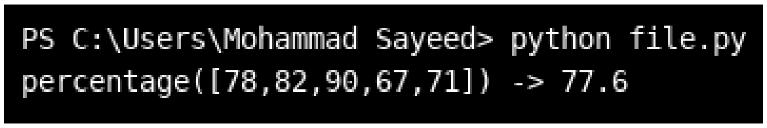
Create a function that accepts marks in 5 subjects and returns the percentage.

Code:

```
def percentage(marks):
    if len(marks) != 5:
        raise ValueError("Exactly 5 marks required")
    return sum(marks)/5.0

print("percentage([78,82,90,67,71]) ->",
percentage([78,82,90,67,71]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
percentage([78,82,90,67,71]) -> 77.6
```

12. Area calculations

Section: User-defined functions

Problem Statement:

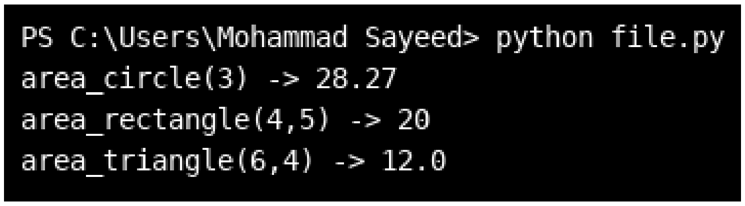
Define a function to calculate the area of a circle, rectangle, and triangle (use parameters).

Code:

```
import math
def area_circle(r):
    return math.pi * r * r
def area_rectangle(l,w):
    return l*w
def area_triangle(b,h):
    return 0.5 * b * h

print("area_circle(3) ->", round(area_circle(3),2))
print("area_rectangle(4,5) ->", area_rectangle(4,5))
print("area_triangle(6,4) ->", area_triangle(6,4))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
area_circle(3) -> 28.27
area_rectangle(4,5) -> 20
area_triangle(6,4) -> 12.0
```

13. Unique elements from list

Section: User-defined functions

Problem Statement:

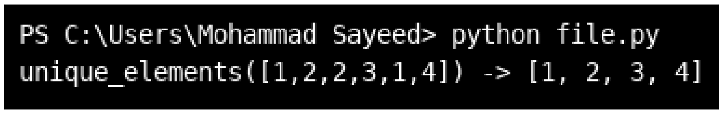
Write a function that takes a list and returns a new list with only unique elements.

Code:

```
def unique_elements(lst):
    return list(dict.fromkeys(lst))

print("unique_elements([1,2,2,3,1,4]) ->",
unique_elements([1,2,2,3,1,4]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
unique_elements([1,2,2,3,1,4]) -> [1, 2, 3, 4]
```

14. Multiplication table

Section: User-defined functions

Problem Statement:

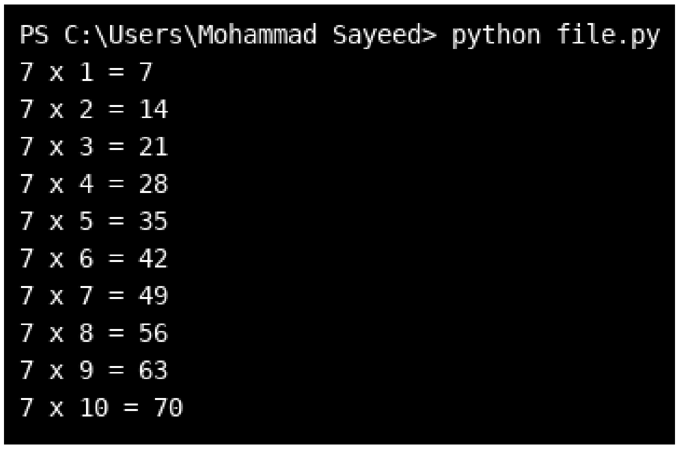
Create a function that accepts a number and prints its multiplication table.

Code:

```
def print_table(n):
    for i in range(1,11):
        print(f"{n} x {i} = {n*i}")

print_table(7)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
```

15. Element-wise sum of two lists

Section: User-defined functions

Problem Statement:

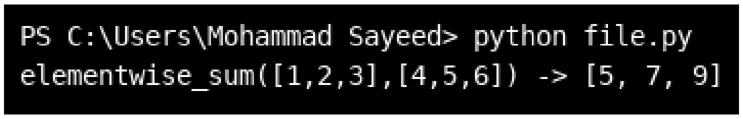
Define a function that takes two lists and returns their element-wise sum.

Code:

```
def elementwise_sum(a,b):
    return [x+y for x,y in zip(a,b)]

print("elementwise_sum([1,2,3],[4,5,6]) ->",
elementwise_sum([1,2,3],[4,5,6]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
elementwise_sum([1,2,3],[4,5,6]) -> [5, 7, 9]
```

16. Add two numbers (lambda)

Section: Lambda functions

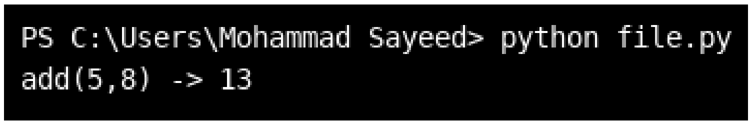
Problem Statement:

Write a lambda function to add two numbers.

Code:

```
add = lambda x,y: x+y
print("add(5,8) ->", add(5,8))
```

Output:


```
PS C:\Users\Mohammad Sayeed> python file.py
add(5,8) -> 13
```

17. Square (lambda)

Section: Lambda functions

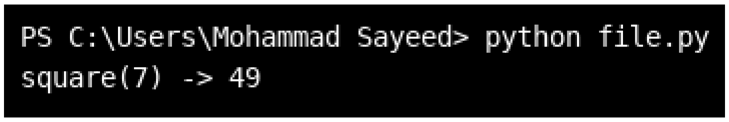
Problem Statement:

Create a lambda to find the square of a number.

Code:

```
square = lambda x: x*x
print("square(7) ->", square(7))
```

Output:


```
PS C:\Users\Mohammad Sayeed> python file.py
square(7) -> 49
```

18. Even or odd (lambda)

Section: Lambda functions

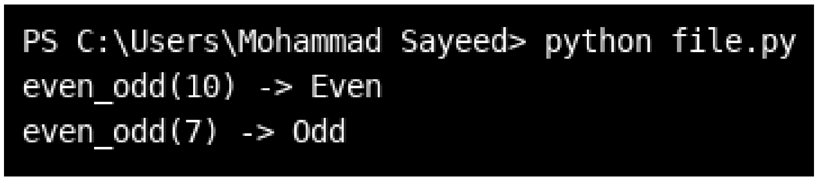
Problem Statement:

Write a lambda that checks whether a number is even or odd.

Code:

```
even_odd = lambda x: 'Even' if x%2==0 else 'Odd'
print("even_odd(10) ->", even_odd(10))
print("even_odd(7) ->", even_odd(7))
```

Output:


```
PS C:\Users\Mohammad Sayeed> python file.py
even_odd(10) -> Even
even_odd(7) -> Odd
```

19. Map double (lambda)

Section: Lambda functions

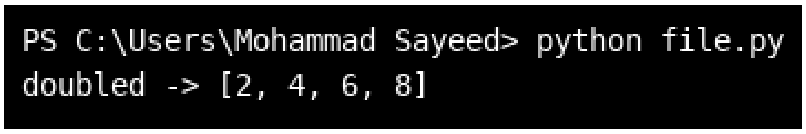
Problem Statement:

Use map() with lambda to double all elements in a list.

Code:

```
nums = [1,2,3,4]
doubled = list(map(lambda x: x*2, nums))
print("doubled ->", doubled)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
doubled -> [2, 4, 6, 8]
```

20. Filter evens (lambda)

Section: Lambda functions

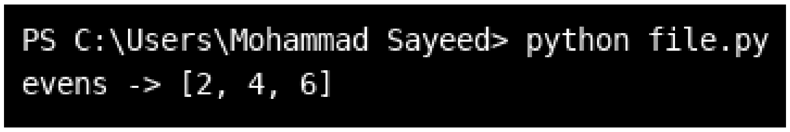
Problem Statement:

Use filter() with lambda to extract only even numbers from a list.

Code:

```
nums = [1,2,3,4,5,6]
evens = list(filter(lambda x: x%2==0, nums))
print("evens ->", evens)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
evens -> [2, 4, 6]
```

21. Reduce product (lambda)
Section: Lambda functions

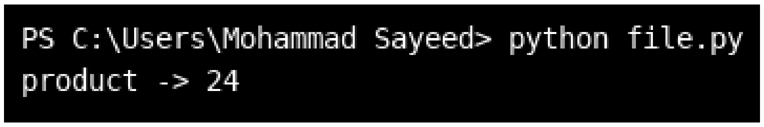
Problem Statement:

Use reduce() (from functools) with lambda to find the product of all numbers in a list.

Code:

```
from functools import reduce
nums = [1,2,3,4]
prod = reduce(lambda a,b: a*b, nums)
print("product ->", prod)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
product -> 24
```

22. Max of two (lambda)
Section: Lambda functions

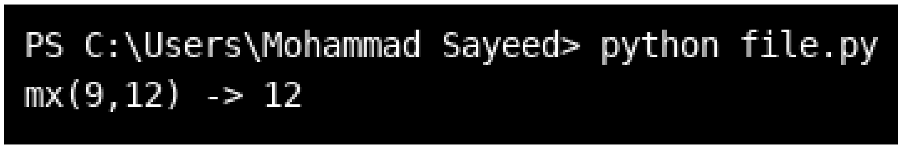
Problem Statement:

Create a lambda that returns the maximum of two numbers.

Code:

```
mx = lambda a,b: a if a>b else b
print("mx(9,12) ->", mx(9,12))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
mx(9,12) -> 12
```

23. Reverse string (lambda)

Section: Lambda functions

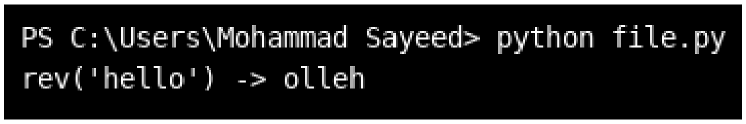
Problem Statement:

Write a lambda to reverse a string.

Code:

```
rev = lambda s: s[::-1]
print("rev('hello') ->", rev('hello'))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
rev('hello') -> olleh
```

24. Sort list of tuples by second element (lambda)

Section: Lambda functions

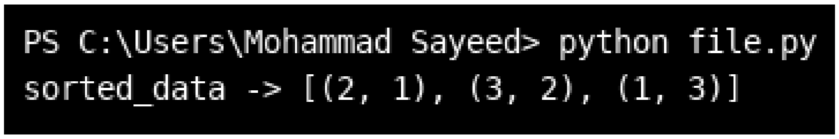
Problem Statement:

Use a lambda to sort a list of tuples based on the second element.

Code:

```
data = [(1,3),(2,1),(3,2)]
sorted_data = sorted(data, key=lambda x: x[1])
print("sorted_data ->", sorted_data)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
sorted_data -> [(2, 1), (3, 2), (1, 3)]
```

25. Uppercase all strings (lambda)

Section: Lambda functions

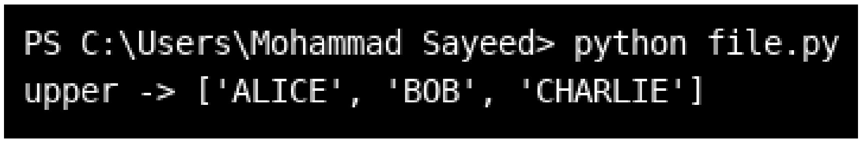
Problem Statement:

Write a lambda function that converts all strings in a list to uppercase.

Code:

```
names = ['alice','Bob','charlie']
upper = list(map(lambda s: s.upper(), names))
print("upper ->", upper)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
upper -> ['ALICE', 'BOB', 'CHARLIE']
```

26. Cube (lambda)

Section: Lambda functions

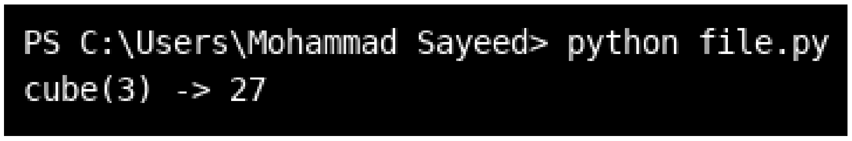
Problem Statement:

Create a lambda to find the cube of a number.

Code:

```
cube = lambda x: x**3
print("cube(3) ->", cube(3))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
cube(3) -> 27
```

27. Starts with vowel (lambda)

Section: Lambda functions

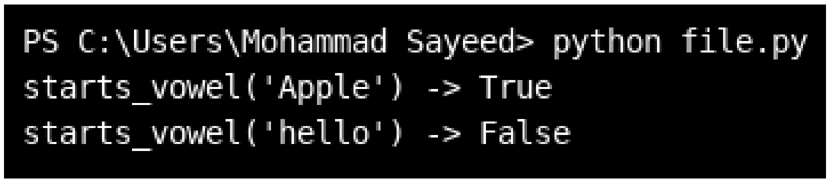
Problem Statement:

Write a lambda to check whether a string starts with a vowel.

Code:

```
starts_vowel = lambda s: s[0].lower() in 'aeiou' if s else False
print("starts_vowel('Apple') ->", starts_vowel('Apple'))
print("starts_vowel('hello') ->", starts_vowel('hello'))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
starts_vowel('Apple') -> True
starts_vowel('hello') -> False
```

28. Celsius to Fahrenheit list (map+lambda)

Section: Lambda functions

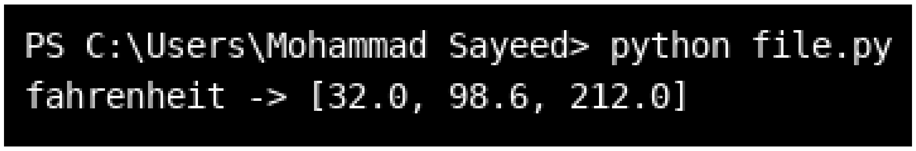
Problem Statement:

Use map() and lambda to convert a list of temperatures from Celsius to Fahrenheit.

Code:

```
celsius = [0,37,100]
f = list(map(lambda c: (c*9/5)+32, celsius))
print("fahrenheit ->", f)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
fahrenheit -> [32.0, 98.6, 212.0]
```

29. Filter names longer than 5 (lambda)

Section: Lambda functions

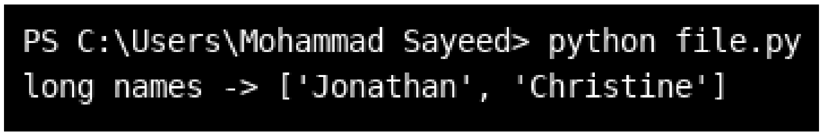
Problem Statement:

Use filter() and lambda to find names longer than 5 characters in a list.

Code:

```
names = ['Alice','Jonathan','Bob','Christine']
long = list(filter(lambda s: len(s)>5, names))
print("long names ->", long)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
long names -> ['Jonathan', 'Christine']
```

30. Square root of numbers (map+lambda)

Section: Lambda functions

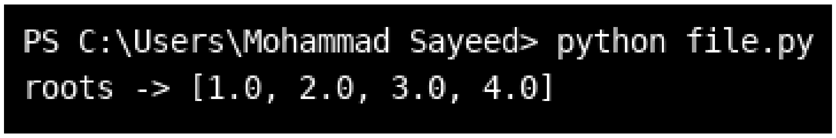
Problem Statement:

Use a lambda and map() to calculate the square root of numbers in a list.

Code:

```
import math
nums = [1,4,9,16]
roots = list(map(lambda x: math.sqrt(x), nums))
print("roots ->", roots)
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
roots -> [1.0, 2.0, 3.0, 4.0]
```

31. Factorial (recursive)

Section: Recursion

Problem Statement:

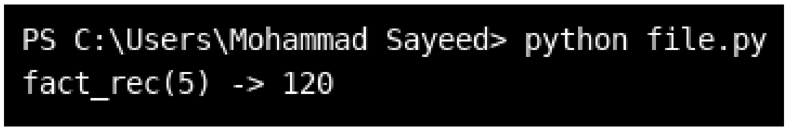
Write a recursive function to calculate the factorial of a number.

Code:

```
def fact_rec(n):
    if n<=1:
        return 1
    return n * fact_rec(n-1)

print("fact_rec(5) ->", fact_rec(5))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
fact_rec(5) -> 120
```

32. Fibonacci series (recursive)

Section: Recursion

Problem Statement:

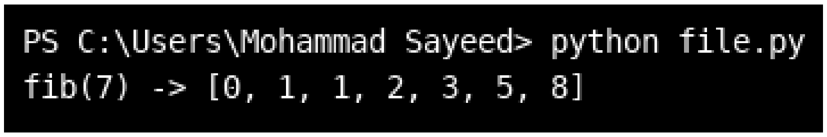
Write a recursive function to generate the Fibonacci series up to n terms.

Code:

```
def fib(n):
    if n<=0: return []
    if n==1: return [0]
    if n==2: return [0,1]
    seq = fib(n-1)
    seq.append(seq[-1] + seq[-2])
    return seq

print("fib(7) ->", fib(7))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
fib(7) -> [0, 1, 1, 2, 3, 5, 8]
```

33. Sum of natural numbers up to n (recursive)

Section: Recursion

Problem Statement:

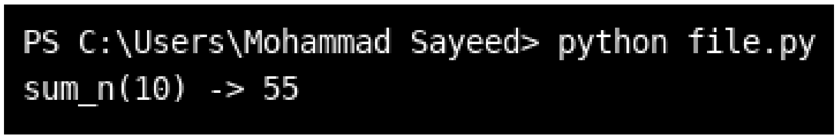
Create a recursive function to find the sum of natural numbers up to n.

Code:

```
def sum_n(n):
    if n==0: return 0
    return n + sum_n(n-1)

print("sum_n(10) ->", sum_n(10))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
sum_n(10) -> 55
```

34. Reverse string (recursive)

Section: Recursion

Problem Statement:

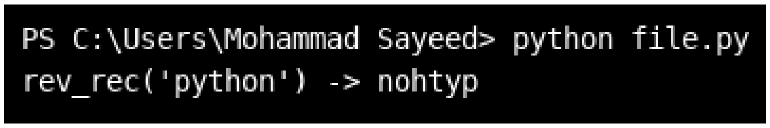
Write a recursive function to reverse a string.

Code:

```
def rev_rec(s):
    if s=="": return ""
    return rev_rec(s[1:]) + s[0]

print("rev_rec('python') ->", rev_rec('python'))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
rev_rec('python') -> nohtyp
```

35. Count digits in number (recursive)

Section: Recursion

Problem Statement:

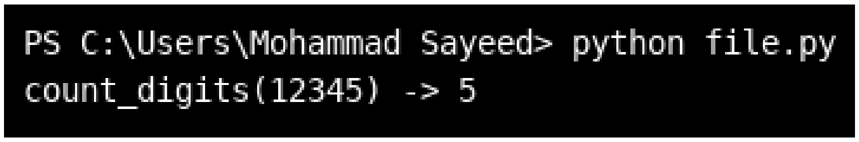
Write a recursive function to count the digits in a number.

Code:

```
def count_digits(n):
    n = abs(n)
    if n < 10: return 1
    return 1 + count_digits(n//10)

print("count_digits(12345) ->", count_digits(12345))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
count_digits(12345) -> 5
```

36. Power calculation x^y (recursive)

Section: Recursion

Problem Statement:

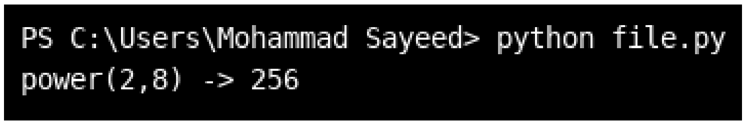
Create a recursive function to calculate the power of a number (x^y).

Code:

```
def power(x,y):
    if y==0: return 1
    if y<0: return 1/power(x,-y)
    return x * power(x,y-1)

print("power(2,8) ->", power(2,8))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
power(2,8) -> 256
```

37. GCD (recursive)

Section: Recursion

Problem Statement:

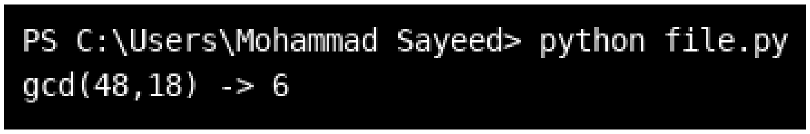
Write a recursive function to find the greatest common divisor (GCD) of two numbers.

Code:

```
def gcd(a,b):
    if b==0: return a
    return gcd(b, a%b)

print("gcd(48,18) ->", gcd(48,18))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
gcd(48,18) -> 6
```

38. Palindrome number (recursive)

Section: Recursion

Problem Statement:

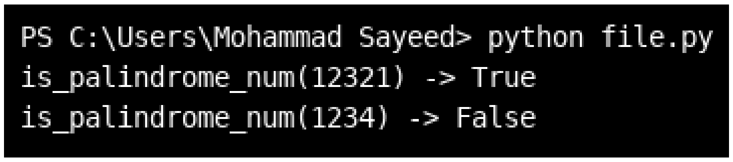
Create a recursive function to check whether a number is palindrome or not.

Code:

```
def is_palindrome_num(n):
    s = str(n)
    return s == s[::-1]

print("is_palindrome_num(12321) ->", is_palindrome_num(12321))
print("is_palindrome_num(1234) ->", is_palindrome_num(1234))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
is_palindrome_num(12321) -> True
is_palindrome_num(1234) -> False
```

39. Sum of elements in list (recursive)

Section: Recursion

Problem Statement:

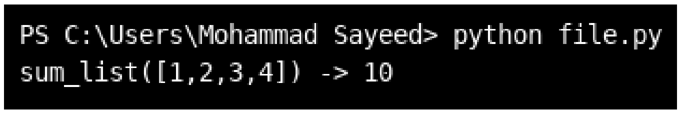
Write a recursive function to find the sum of elements in a list.

Code:

```
def sum_list(lst):
    if not lst: return 0
    return lst[0] + sum_list(lst[1:])

print("sum_list([1,2,3,4]) ->", sum_list([1,2,3,4]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
sum_list([1,2,3,4]) -> 10
```

40. Print list in reverse (recursive)

Section: Recursion

Problem Statement:

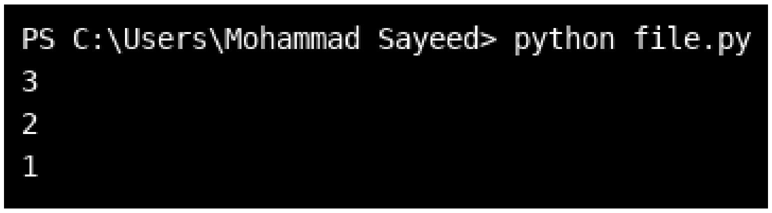
Write a recursive function to print all elements of a list in reverse order.

Code:

```
def print_reverse(lst):
    if not lst: return
    print(lst[-1])
    print_reverse(lst[:-1])

print_reverse([1,2,3])
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
3
2
1
```

41. Length of string (recursive)

Section: Recursion

Problem Statement:

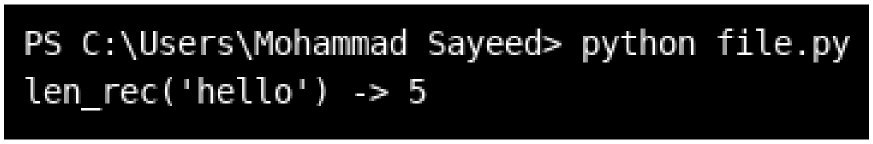
Create a recursive function to find the length of a string.

Code:

```
def len_rec(s):
    if s=='': return 0
    return 1 + len_rec(s[1:])

print("len_rec('hello') ->", len_rec('hello'))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
len_rec('hello') -> 5
```

42. Decimal to binary (recursive)

Section: Recursion

Problem Statement:

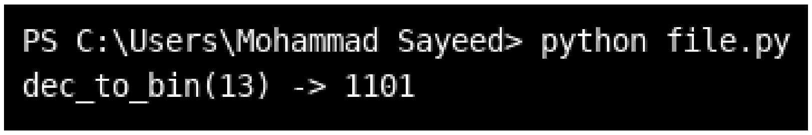
Write a recursive function to convert a decimal number to binary.

Code:

```
def dec_to_bin(n):
    if n==0: return '0'
    if n==1: return '1'
    return dec_to_bin(n//2) + str(n%2)

print("dec_to_bin(13) ->", dec_to_bin(13))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
dec_to_bin(13) -> 1101
```

43. Minimum element in list (recursive)

Section: Recursion

Problem Statement:

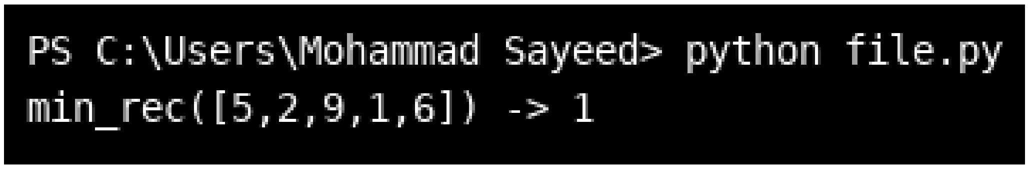
Create a recursive function to find the minimum element in a list.

Code:

```
def min_rec(lst):
    if len(lst)==1: return lst[0]
    m = min_rec(lst[1:])
    return lst[0] if lst[0] < m else m

print("min_rec([5,2,9,1,6]) ->", min_rec([5,2,9,1,6]))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
min_rec([5,2,9,1,6]) -> 1
```

44. Sum of digits (recursive)

Section: Recursion

Problem Statement:

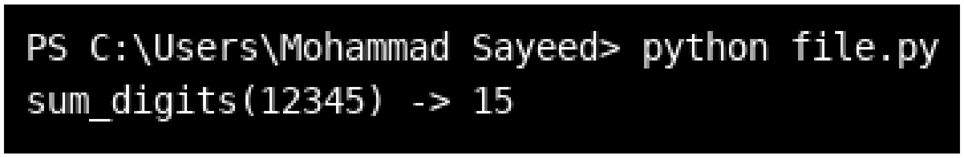
Write a recursive function to compute the sum of digits of a number.

Code:

```
def sum_digits(n):
    n = abs(n)
    if n < 10: return n
    return n%10 + sum_digits(n//10)

print("sum_digits(12345) ->", sum_digits(12345))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
sum_digits(12345) -> 15
```

45. String palindrome (recursive)

Section: Recursion

Problem Statement:

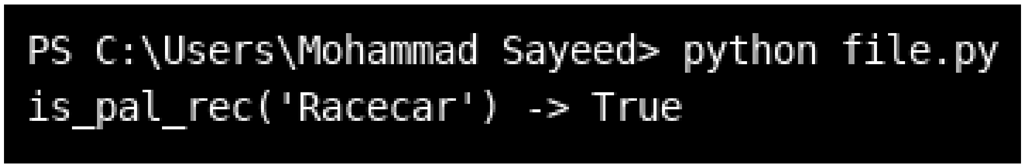
Create a recursive function to check whether a given string is palindrome.

Code:

```
def is_pal_rec(s):
    s2 = ''.join(ch.lower() for ch in s if ch.isalnum())
    def helper(l,r):
        if l>=r: return True
        if s2[l]!=s2[r]: return False
        return helper(l+1,r-1)
    return helper(0,len(s2)-1)

print("is_pal_rec('Racecar') ->", is_pal_rec('Racecar'))
```

Output:



```
PS C:\Users\Mohammad Sayeed> python file.py
is_pal_rec('Racecar') -> True
```