6th-july-ppt-ass-1

July 7, 2023

0.1 1. Write a Python program to reverse a string without using any built-in string reversal functions.

[7]:
```
str = "my name is pradeep singh rajput and i am pursuing a masters course at␣
 ↪iit kharagpur"
print("reverse string is : ",str[::-1])
```

```
reverse string is :  rupgarahk tii ta esruoc sretsam a gniusrup ma i dna tupjar
hgnis peedarp si eman ym
```

0.2 2. Implement a function to check if a given string is a palindrome.

[6]:
```
def is_palindrome(string):
    left = 0
    right = len(string)-1
    while left<right:
        if string[left]!=string[right]:
            return False
        else:
            string[left]==string[right]
            left +=1
            right-=1

    return True
string = 'pradeep'
result = is_palindrome(string)
print("Given string is a palindrome :" , result)
```

```
Given string is a palindrome : False
```

0.3 3. Write a program to find the largest element in a given list.

[1]:
```
mylist = [12,4,45,98,45]
largest_element = max(mylist)
print("largest elemnets in above list is :" , largest_element)
```

```
largest elemnets in above list is : 98
```

0.4 4. Implement a function to count the occurrence of each element in a list.

[18]:
```
import pandas as pd

mylist = [2,3,2,3,4,5,6]
count = pd.Series(mylist).value_counts()
print("ele_cnt")
print(count)
```

```
ele_cnt
2    2
3    2
4    1
5    1
6    1
dtype: int64
```

0.5 5. Write a Python program to find the second largest number in a list.

[ ]:
```

```

0.6 6. Implement a function to remove duplicate elements from a list.

[28]:
```
def remove_duplicate(input_list):
    unique_item = []
    for num in input_list:
        if num not in unique_item:
            unique_item.append(num)

    return  unique_item

input_list = [2,3,2,3,4,5,6]
result = remove_duplicate(input_list)
print("unique numbers are in the input list:", result)
```

```
unique numbers are in the input list: [2, 3, 4, 5, 6]
```

[31]:
```
mylist = [2,3,2,3,4,5,6]
list(set(mylist))
```

[31]:
```
[2, 3, 4, 5, 6]
```

0.7 7. Write a program to calculate the factorial of a given number.

[22]:
```
def findfactorial(num):
    if (num==0 or num ==1):
        return 1

    return num*findfactorial(num-1)    ### n! = n*(n-1)!

num = 1
result =  findfactorial(num)
print("factorial of given number is :", result)
```

```
factorial of given number is : 1
```

0.8 8. Implement a function to check if a given number is prime.

[27]:
```
def isprimenumber(n):
    if n <2:
        return False

    for i in range(2,n):
        if n%i==0:
            return False

    return True

n = 15
result = isprimenumber(n)
print("Given number is a prime number:", result )
```

```
Given number is a prime number: False
```

0.9 9. Write a Python program to sort a list of integers in ascending order.

[20]:
```
lst = [12,89,45,13,56,78]
results  = sorted(lst)
print("Sorted in ascending order above list is :", results)
```

```
Sorted in ascending order above list is : [12, 13, 45, 56, 78, 89]
```

0.10 10. Implement a function to find the sum of all numbers in a list.

[18]:
```
lst = [12,89,45,13,56,78]
results  = sum(lst)
print("Sum of the all elements present in the list is : ", results)
```

```
Sum of the all elements present in the list is :  293
```

[ ]:
```
# 11. Write a program to find the common elements between two lists.
```

[31]:
```
print("Commmon elements between list1 and list2 are :")
list1 = [12,89,45,13,56,78]
list2 =  [2,89,54,65,78,55]
lst = []
for i in list1 :
    if i  in list2:
        lst.append(i)
        print(i)
```

```
Commmon elements between list1 and list2 are :
89
78
```

[34]:
```
## Method 2 - using def function
def find_commen_element(list1,list2):
    commonelementslist = []
    for i in list1:
        if i in list2:
            commonelementslist.append(i)
    return commonelementslist

list1 = [12,89,45,13,56,78]
list2 =  [2,89,54,65,78,55]
results = find_commen_element(list1,list2)
print("Commmon elements between list1 and list2 are :",results)
```

```
Commmon elements between list1 and list2 are : [89, 78]
```

0.11 12. Implement a function to check if a given string is an anagram of another string.

[31]:
```
def is_anagram(string1,string2):
    sorted_str1 = sorted(string1)
    sorted_str2 = sorted(string2)
    if len(sorted_str1)==len(sorted_str2):
         print("Yes they are an anagram")




string1 = "pradeep singh".replace(" ","").lower()
string2 = 'singh pradeep'.replace(" ","").lower()
result = is_anagram(string1,string2)
```

```
Yes they are an anagram
```

0.12 13. Write a Python program to generate all permutations of a given string.

[ ]:
```
```

0.13 14. Implement a function to calculate the Fibonacci sequence up to a given number of terms.

[17]:
```
def cal_fibonacci(n):
    a = 0
    b = 1
    if n<0:
        print("please enter positive number")
    elif n ==0:
        return a
    elif n==1:
        return b
    else:
        for i in range(2,n+1):
            c = a+b
            a = b
            b = c
    return b
cal_fibonacci(10)
```

[17]:
```
55
```

0.14 15. Write a program to find the median of a list of numbers.

[6]:
```
import numpy as np
def find_median(list1):
    median = np.median(list1)
    return median


list1 = [12,89,45,13,56,78]
results = find_median(list1)
print("median of sorted array is ", results)
```

```
median of sorted array is  50.5
```

0.15 16. Implement a function to check if a given list is sorted in non-decreasing order.

[12]:
```
nums = [12,89,45,13,56,78]
results  = sorted(nums)
print("Sorted in ascending order above list is :", results)
```

```
Sorted in ascending order above list is : [12, 13, 45, 56, 78, 89]
```

0.16 17. Write a Python program to find the intersection of two lists.

[11]:
```python
def intersection(list1,list2):
    return list(set(list1)&set(list2))


list1 = [12,89,45,13,56,78]
list2 =  [2,89,54,65,78,55]
results = intersection(list1,list2)
print("Intersection between list1 and list2 are :",results)
```

```
Intersection between list1 and list2 are : [89, 78]
```

0.17 18. Implement a function to find the maximum subarray sum in a given list.

[ ]:
```python

```

0.18 19. Write a program to remove all vowels from a given string.

[25]:
```python
def remove_vowels(str):
    vowels = 'aieouAIEOU'
    result = ""
    for char in str:
        if char not in vowels:
            result +=char

    return result


str = "Write a program to remove all vowels from a given string"
print("vowels removed string is :" ,remove_vowels(str))
```

```
vowels removed string is : Wrt  prgrm t rmv ll vwls frm  gvn strng
```

0.19 20. Implement a function to reverse the order of words in a given sentence

[34]:
```python
def reverse_sentence(sentence):
    words = sentence.split()
    reversed_words = words[::-1]
    reversed_sentence = ' '.join(reversed_words)
    return reversed_sentence

# Example usage
```

```
sentence = "Implement a function to reverse the order of words in a given␣
 ↪sentence"
reversed_sentence = reverse_sentence(sentence)
print("Reversed sentence:", reversed_sentence)
```

```
Reversed sentence: sentence given a in words of order the reverse to function a
Implement
```

0.20 21. Write a Python program to check if two strings are anagrams of each other.

[40]:
```
def is_anagram(string1,string2):
    sorted_str1 = sorted(string1)
    sorted_str2 = sorted(string2)
    if len(sorted_str1)==len(sorted_str2):
         print("Yes they are an anagram")

string1 = "pradeep singh".replace(" ","").lower()
string2 = 'singh pradeep'.replace(" ","").lower()
result = is_anagram(string1,string2)
```

```
Yes they are an anagram
```

0.21 22. Implement a function to find the first non-repeating character in a string.

0.22 23. Write a program to find the prime factors of a given number.

0.23 24. Implement a function to check if a given number is a power of two.

[39]:
```
def is_power_of_two(number):
    if number <= 0:
        return False
    elif number == 1:
        return True
    else:
        return (number & (number - 1)) == 0

# Example usage
num = 15

if is_power_of_two(num):
    print(num, "is a power of two.")
else:
    print(num, "is not a power of two.")
```

```
15 is not a power of two.
```

0.24 25. Write a Python program to merge two sorted lists into a single sorted list.

[47]:
```
list1 = [12,89,45,13,56,78]
list2 =  [2,89,54,65,78,55]
sorted_list1 = sorted(list1)
sorted_list2 = sorted(list2)
single_sorted_list = sorted_list1+sorted_list2
single_sorted_list
```

[47]: [12, 13, 45, 56, 78, 89, 2, 54, 55, 65, 78, 89]

[54]:
```
list1 = [12,89,45,13,56,78]
list2 =  [2,89,54,65,78,55]
sorted_list1 = sorted(list1)
sorted_list2 = sorted(list2)
sorted_list1.extend(sorted_list2)
print("Sorted merge list is :",sorted_list1)
```

```
Sorted merge list is : [12, 13, 45, 56, 78, 89, 2, 54, 55, 65, 78, 89]
```

0.25 26. Implement a function to find the mode of a list of numbers.

[53]:
```
import statistics
def find_mode(numbers):
    modes = statistics.multimode(numbers)
    return modes



numbers = [1,4,5,5,4,6,6,2,4,3]
result = find_mode(numbers)
print("mode of a given list is :",result)
```

```
mode of a given list is : [4]
```

0.26 27. Write a program to find the greatest common divisor (GCD) of two numbers.

[57]:
```
import math
num1 = 15
num2 = 105
gcd = math.gcd(num1,num2)

print("The greatest common divisor (GCD) of two numbers", gcd)
```

```
The greatest common divisor (GCD) of two numbers 15
```

[59]:
```
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a

a = 48
b = 180

gcd_value = gcd(a, b)
print(gcd_value)
```

```
12
```

0.27 28. Implement a function to calculate the square root of a given number.

[59]:
```
def cal_square_root(x):
    return x**0.5

x = 27
results = cal_square_root(x)
print("square root of a given number is :",results)
```

```
square root of a given number is : 5.196152422706632
```

[61]:
```
num = int(input("enter a number ?"))
num**0.5
```

```
enter a number ? 27
```

[61]:
```
5.196152422706632
```

[ ]:
```
29. Write a Python program to check if a given string is a valid palindrome␣
↪ignoring non-alphanumeric characters.
```

[62]:
```
# 30. Implement a function to find the minimum element in a rotated sorted list.
def find_min_element(list):
    minmun_number = min(list)
    return minmun_number

list  = [4, 5, 6, 7, 1, 2, 3]
results = find_min_element(list)
print("The minimum element in a rotated sorted list is :" ,results)
```

```
The minimum element in a rotated sorted list is : 1
```

[63]:
```
list  = [4, 5, 6, 7, 1, 2, 3]
min(list)
```

[63]: 1

0.28 31. Write a program to find the sum of all even numbers in a list.

[ ]:
```
def sum_all_even_nums(numbers):
    sum_even = 0
    for num in numbers:
        if num%2==0:
            sum_even += num
    return sum_even
numbers = [12,89,45,13,56,78]
result = sum_all_even_nums(numbers)
print("The the sum of all even numbers in a list is :" ,result)
```

0.29 32. Implement a function to calculate the power of a number using recursion.

[60]:
```
def power(base,exponent):
    if exponent ==0:
        return 1
    else:
        return base * power(base, exponent - 1)

base  = 2
exponent = 10
result = power(base,exponent)
print("The power of a given number is : " , result)
```

```
The power of a given number is :  1024
```

0.30 33. Write a Python program to remove duplicates from a list while preserving the order.

0.31 34. Implement a function to find the longest common prefix among a list of strings.

0.32 35. Write a program to check if a given number is a perfect square.

[ ]:
```
n = 2500
for i in range(n+1):
    if i**2==n:
        print("yes")
        break
else:
    print("No")
```

0.33 36. Implement a function to calculate the product of all elements in a list.

[16]:
```
def find_sum_digits(number):
    str_num = str(number)
    sum_digits = 1
    for digit_char in str_num:
        digit = int(digit_char)
        sum_digits *=digit
    return sum_digits


number = 956
result = find_sum_digits(number)
print("sum of digits of a given number is :", result)
```

```
sum of digits of a given number is : 270
```

0.34 37. Write a Python program to reverse the order of words in a sentence while preserving the word order.

[1]:
```
def reverse_sentence(sentence):
    words = sentence.split()
    reversed_words = words[::-1]
    reversed_sentence = ' '.join(reversed_words)
    return reversed_sentence

# Example usage
sentence = "Implement a function to reverse the order of words in a given␣
 ↪sentence"
reversed_sentence = reverse_sentence(sentence)
print("Reversed sentence:", reversed_sentence)
```

```
Reversed sentence: sentence given a in words of order the reverse to function a
Implement
```

0.35 38. Implement a function to find the missing number in a given list of consecutive numbers

[3]:
```
def find_missing_num(numbers):
    sorted_number = sorted(numbers)
    n = len(numbers)+1
    expected_sum = n*(n+1)/2
    ectual_sum = sum(numbers)
    missing_value = expected_sum - ectual_sum
    return missing_value


numbers = [1,4,5,6,2,7]
```

```
result = find_missing_num(numbers)
print("The missing number in a given list of consecutive numbers is : ", result)
```

```
The missing number in a given list of consecutive numbers is :  3.0
```

0.36 39. Write a program to find the sum of digits of a given number.

[14]:
```
def find_sum_digits(number):
    str_num = str(number)
    sum_digits = 0
    for digit_char in str_num:
        digit = int(digit_char)
        sum_digits +=digit
    return sum_digits


number = 956
result = find_sum_digits(number)
print("sum of digits of a given number is :", result)
```

```
sum of digits of a given number is : 20
```

[ ]:
```

```

0.37 40. Implement a function to check if a given string is a valid palindrome considering case sensitivity.

[ ]:
```

```

0.38 41. Write a Python program to find the smallest missing positive integer in a list.

[ ]:
```

```

0.39 42. Implement a function to find the longest palindrome substring in a given string

[ ]:
```

```

0.40 43. Write a program to find the number of occurrences of a given element in a list

[59]:
```
mylist = [2,3,2,3,4,2,3,2,3,5,6]
mylist.count(2)
```

[59]:
```
4
```

[58]:
```
mylist.count(3)
```

[58]: 4

0.41 44. Implement a function to check if a given number is a perfect number.

[32]:
```
def isperfectnumber(number):
    sum_divisors = 0
    for i in range(1,number//2+1):
        if number%i==0:
            sum_divisors+=i

    return sum_divisors==number

number = 6
result = isperfectnumber(number)
print("Given number is perfect number :" ,result)
```

```
Given number is perfect number : True
```

0.42 45. Write a Python program to remove all duplicates from a string.

[53]:
```
#my_str = "pradeep pradeep singh singh"
def remove_duplicate(my_str):
    unique_str = []
    for char in my_str:
        if char not in unique_str:
            unique_str.append(char)

    return  ''.join(unique_str)
my_str = "hello pradeep singh rajput"
result = remove_duplicate(my_str)
print("unique characters are in the my string:", result)
```

```
unique characters are in the my string: helo pradsingjut
```

[45]:
```
def remove_duplicates(string):
    unique_chars = []
    for char in string:
        if char not in unique_chars:
            unique_chars.append(char)
    return ''.join(unique_chars)

# Example usage
input_string = "Hello, World!"
result = remove_duplicates(input_string)
print("String with duplicates removed:", result)
```

```
String with duplicates removed: Helo, Wrd!
```

[ ]: