Table of Contents

**Basic programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Number programs**

Type chapter title (level 2)

Type chapter title (level 2)

Type chapter title (level 2)

**String programs**

[1. Python program to find common element in given 2 strings](#_Python_program_to_6)

[2. Python program to find frequency/count of words in given sentence](#_Python_program_to_7)

Type chapter title (level 2)

Type chapter title (level 2)

Type chapter title (level 2)

Type chapter title (level 2)

[**List programs**](#_List_programs:)

[1.Python program to find largest number in a list](#_Python_program_to)

[2. Python program to find second largest number in a list](#_Python_program_to_1)

[3.Python program to find least/small number in a list](#_Python_program_to_2)

[4.Python program to find second least/small number in a list](#_Python_program_to_3)

[5. Python program to find largest & smallest length string from given list](#_Python_program_to_4)

[6.Python program to find N largest elements from a list](#_Python_program_to_5)

[7. Python program to find missing number from congestive list of numbers.](#_Python_program_to_11)

Type chapter title (level 2)

**Tuple programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Set programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Dictionary programs**

[1. Python program to convert 2 lists into dict.](#_Python_program_to_9)

Type chapter title (level 2)

Type chapter title (level 2)

Type chapter title (level 2)

Type chapter title (level 2)

**Functions programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Pattern Printing programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Oops programs**

Type chapter title (level 2)

Type chapter title (level 3)

**Exception programs**

Type chapter title (level 2)

Type chapter title (level 3)

**File handling programs**

Type chapter title (level 2)

Type chapter title (level 3)

## **Number programs:**

## **String programs:**

### Python program to find common elements in given 2 strings.

|  |  |  |
| --- | --- | --- |
| Input : s1 = REENE ; s2=NAINA  Output : N  **Solution:**   |  | | --- | | 🡪 Using **intersection()** built-in function:  str1 = "REENE" str2="NAINA" s1 = set(str1) s2 = set(str2) print(\*(s1 & s2))  Output:  N | | 🡪 Using naive method:  str1 = "REENE" str2="NAINA" common\_ele = [] s1 = set(str1) s2 = set(str2) for i in s1:  if i in s2:  common\_ele.append(i) print(\*common\_ele)  Output:  N | |

### Python program to find frequency/count of words in given sentence.

|  |  |  |
| --- | --- | --- |
| Input : “Sheena loves eating apple and mango Her sister also loves eating apple and mango”  Output : {'Sheena': 1, 'loves': 2, 'eating': 2, 'apple': 2, 'and': 2, 'mango': 2, 'Her': 1, 'sister': 1, 'also': 1}  **Solution:**   |  | | --- | | 🡪 Using **if & else** built-in function:  s = "Sheena loves eating apple and mango Her sister also loves eating apple and mango" words = s.split(" ") d = {} for word in words:  if word not in d:  d[word] = 1  else:  d[word] +=1 print(d)  Output:  {'Sheena': 1, 'loves': 2, 'eating': 2, 'apple': 2, 'and': 2, 'mango': 2, 'Her': 1, 'sister': 1, 'also': 1} | | 🡪 Using if only method:  s = "Sheena loves eating apple and mango Her sister also loves eating apple and mango" words = s.split(" ") d = {} for word in words:  if word not in d:  d[word] = 0  d[word] +=1 print(d)  Output:  {'Sheena': 1, 'loves': 2, 'eating': 2, 'apple': 2, 'and': 2, 'mango': 2, 'Her': 1, 'sister': 1, 'also': 1} | |

## **List programs:**

### Python program to find largest number in a list

|  |  |  |
| --- | --- | --- |
| Input : [10, 20, 4, 45, 99]  Output : 99  **Solution:**   |  | | --- | | 🡪 Using **sorted** built-in function:  l = [10, 20, 4, 45, 99] print(sorted(l)[-1]) print(sorted(l,reverse=True)[0])  Output:  99  99 | | 🡪 Using naive method:  l = [10, 20, 4, 45, 99] largest = l[0] for number in l:  if number>largest:  largest = number print(largest)  Output:  99 | |

### Python program to find second largest number in a list

|  |  |  |
| --- | --- | --- |
| Input : [10, 20, 4, 45, 99]  Output : 45  **Solution:**   |  | | --- | | 🡪 Using **sorted** built-in function:  l = [10, 20, 4, 45, 99] print(sorted(l)[-2]) print(sorted(l,reverse=True)[1])  Output:  45  45 | | 🡪 Using naive method:  l = [10, 20, 4, 45, 99] largest = second = l[0] for number in l:  if number>largest:  second = largest  largest = number  elif largest>number>second:  second = number print(second)  Output:  45 | |

### Python program to find least/small number in a list

|  |  |  |
| --- | --- | --- |
| Input : [10, 20, 4, 45, 99]  Output : 4  **Solution:**   |  | | --- | | 🡪 Using **sorted** built-in function:  l = [10, 20, 4, 45, 99] print(sorted(l)[0]) print(sorted(l,reverse=True)[-1])  Output:  4  4 | | 🡪 Using naive method:  l = [10, 20, 4, 45, 99] least = l[0] for number in l:  if number<least:  least = number print(least)  Output:  4 | |

### Python program to find second least/small number in a list

|  |  |  |
| --- | --- | --- |
| Input : [10, 20, 4, 45, 99]  Output : 10  **Solution:**   |  | | --- | | 🡪 Using **sorted** built-in function:  l = [10, 20, 4, 45, 99] print(sorted(l)[1]) print(sorted(l,reverse=True)[-2])  Output:  10  10 | | 🡪 Using naive method:  l = [10, 20, 4, 45, 99] least = second\_least = l[0] for number in l:  if number<least:  least = number  elif least<number<second\_least:  second\_least = number print(second\_least)  Output:  45 | |

### Python program to find largest & smallest length string from given list

|  |  |  |  |
| --- | --- | --- | --- |
| Input : ["Python","Java","C","C++","Dot net"]  Output : Dot net  **Solution:**   |  | | --- | | 🡪 Using **max** built-in function:  l = ["Python","Java","C","C++","Dot net"] print(max(l,key=lambda x:len(x))) print(min(l,key=lambda x:len(x)))  Output:  Dot net  C | | 🡪 Using **sorted** built-in function:  l = ["Python","Java","C","C++","Dot net"] print(sorted(l,key=lambda x:len(x),reverse=True)[0]) print(sorted(l,key=lambda x:len(x),reverse=True)[-1])  Output:  Dot net  C | | 🡪 Using naive method  l = ["Python","Java","C","C++","Dot net"] largest = l[0] smallest = l[0] for i in l:  if len(largest) <len(i):  largest = i  if len(smallest) >len(i):  smallest = i print(largest) print(smallest)  Output:  Dot net  C | |

### Python program to find N largest elements from a list:

|  |  |  |
| --- | --- | --- |
| **Input :**  l = [2, 6, 41, 85, 0, 3, 7, 6, 10]  n = 3  **Output :**  [85,41,10]  **Solution:**   |  | | --- | | 🡪 Using **sorted** built-in function:  l = [2, 6, 41, 85, 0, 3, 7, 6, 10] n = 3 print(sorted(l,reverse=True)[:n])  Output:  [85, 41, 10] | | 🡪 Using naive method:  l = [2, 6, 41, 85, 0, 3, 7, 6, 10] n = 3 res = []  def largest\_fun(l):  largest = l[0]  for i in l:  if i>largest:  largest = i  return largest  for i in range(n):  largest = largest\_fun(l)  res.append(largest)  l.remove(largest) print(res)  Output:  [85, 41, 10] | |

### Python program to find missing number from congestive list of numbers.

|  |  |  |
| --- | --- | --- |
| Input : [1,2,4,5,6,7]  Output : 3  **Solution:**   |  | | --- | | 🡪 Using **summation** built-in function:  l = [1,2,4,5,6,7] n = len(l)+1 actual\_sum = sum(l) expected\_sum = (n\*(n+1))//2 print(expected\_sum-actual\_sum)  Output:  3 | | 🡪 Using XOR method:  l = [1,2,4,5,6,7] n = len(l) xor\_a = l[0]  for index in range(1,n):  xor\_a = xor\_a ^l[index] x2 = 0 for index in range(1,n+2):  x2 = x2^index print(xor\_a^x2)  Output:  3 | |

## **Dict Programs:**

### Python program to convert 2 lists into dict.

|  |  |  |
| --- | --- | --- |
| Input :  L1 = [Naina,Kimi,Sheena]  L2 = [8,4,5]  Output :  {'Naina': 8, 'Kimi': 4, 'Sheena': 5}  **Solution:**   |  | | --- | | 🡪 Using **Dict comprehension** built-in function:  l1 = ["Naina", "Kimi", "Sheena"] l2 = [8, 4, 5]  *# Convert 2 lists into dict* d = {k:v for k,v in zip(l1,l2)} print(d)  *# Conevert dict into list of tuples* print([i for i in d.items()])  Output:  {'Naina': 8, 'Kimi': 4, 'Sheena': 5}  [('Naina', 8), ('Kimi', 4), ('Sheena', 5)] | | 🡪 Using **Dict constructor** method:  l1 = ["Naina", "Kimi", "Sheena"] l2 = [8, 4, 5]  *# Convert 2 lists into dict* d = dict(zip(l1,l2)) print(d)  *# Conevert dict into list of tuples* print([i for i in d.items()])  Output:  {'Naina': 8, 'Kimi': 4, 'Sheena': 5}  [('Naina', 8), ('Kimi', 4), ('Sheena', 5)] | |