20th_August_Python_Basics_Practice

September 15, 2023

[ ]:
```
#Write a program to reverse a string.
try:

 def reverse(s):
    str = ""
    for i in s:
        str = i + str
    return str


 x = input("Enter the value of string?")

 print("The reversed string(using loops) is : ")
 print(reverse(x))

except:
    logging.info('blah', exc_info=True)
```

[ ]:
```
#Check if a string is a palindrome.
try:

    def isPalindrome(s):
        return s == s[::-1]


    s = "malayalam"
    ans = isPalindrome(s)

    if ans:
        print("Yes")
    else:
        print("No")
except:
    logging.info('blah', exc_info=True)
```

[3]:
```
#Convert a string to uppercase.
try:
    stringVar = "flying"
    print(stringVar.upper())
except:
    logging.info('blah', exc_info=True)
```

```
FLYING
```

[4]:
```
#Convert a string to lowercase.
try:
    stringVar = "Abhi"
    print(stringVar.lower())
except:
    logging.info('blah', exc_info=True)
```

```
abhi
```

[5]:
```
#Count the number of vowels in a string.
try:
 string = input("Please enter your line: ")
 vowels = 0
 for i in string:
     if (i == 'a' or i == 'e' or i == 'i' or i == 'o' or i == 'u' or i == 'A'
             or i == 'E' or i == 'I' or i == 'O' or i == 'U'):
        vowels = vowels + 1
 print("Number of vowels are:")
 print(vowels)
except:
    logging.info('blah', exc_info=True)
```

```
Please enter your line:  i am great
```

```
Number of vowels are:
4
```

[8]:
```
# Python program to count consonant in a string

try:

 def countConsonants(string):
    num_consonants = 0
    # to count the consonants
    for char in string:
        if char not in "aeiouAEIOU ":
            num_consonants += 1
    return num_consonants


 # take input
 string = input('Enter any string: ')

 # calling function and display result
 print('No of consonants:', countConsonants(string))

except:
    logging.info('blah', exc_info=True)
```

```
Enter any string:  Abhishek

No of consonants: 5
```

[9]:
```
#Remove all whitespaces from a string.
try:
    str1 = "         Welcome to Python classes"
    print("The given string is: ",str1)
    print("After removing the leading white spaces")
    print(str1.replace(" ",""))
except:
    logging.info('blah', exc_info=True)
```

```
The given string is:           Welcome to Python classes
After removing the leading white spaces
WelcometoPythonclasses
```

[10]:
```
#Find the length of a string without using the `len()` function.
string=input("Enter string:")
count=0
for i in string:
      count=count+1
print("Length of the string is:")
print(count)
```

```
Enter string: Bharat

Length of the string is:
6
```

[11]:
```
#Check if a string contains a specific word.
string = "Hello, world! Welcome to Python."
word = "world"

if word in string:
    print("The string contains the word.")
else:
    print("The string does not contain the word.")
```

```
The string contains the word.
```

[12]:
```python
#Replace a word in a string with another word.
string = "Good Morning"
new_string = string.replace("Good", "Great")

print(new_string)
```

```
Great Morning
```

[13]:
```python
#Count the occurrences of a word in a string.
def countOccurrences(str, word):



    a = str.split(" ")

    # search for pattern in a
    count = 0
    for i in range(0, len(a)):

        # if match found increase count
        if (word == a[i]):
           count = count + 1

    return count


str ="GeeksforGeeks A computer science portal for geeks  "
word ="for"
print(countOccurrences(str, word))
```

```
1
```

[14]:
```python
#Find the first occurrence of a word in a string.
string = "I love my India."
print(string.find("love"));
```

```
2
```

[1]:
```python
#Find the last occurrence of a word in a string.
s = "the dude is a cool dude"
s.find('dude')
```

[1]:
```
4
```

[2]:
```python
#Find the last occurrence of a word in a string.
test_string ="India is a democratic country and also is a developing country"



tar_word = "is"
```

```
print("The original string : " + str(test_string))


res = test_string.rindex(tar_word)


print("Index of last occurrence of substring is : " + str(res))
```

```
The original string : India is a democratic country and also is a developing
country
Index of last occurrence of substring is : 39
```

[3]:
```
#Split a string into a list of words.
lst =  "I am proud of my country"
print( lst.split())
```

```
['I', 'am', 'proud', 'of', 'my', 'country']
```

[4]:
```
#Join a list of words into a string.
words = ['this', 'is', 'a', 'sentence']
newWord1=' '.join(words)
print(newWord1)
```

```
this is a sentence
```

[5]:
```
#Convert a string where words are separated by spaces to one where words
are separated by underscores.
mystring="m a n o r"
mystring1=mystring.replace(" ", "_")
print(mystring1)
```

```
m_a_n_o_r
```

[8]:
```
#Check if a string starts with a specific word or phrase.

#Check if a string ends with a specific word or phrase.
var = "Gadar Katha"

print(var.startswith("Gadar"))
print(var.endswith("Katha"))
```

```
True
True
```

[9]:
```
#Convert a string to title case (e.g., "hello world" to "Hello World").
name = 'hello world'.title()
```

```
print(name)
```

```
Hello World
```

[10]:
```
#Find the shortest word in a string.
s = 'I am not at all well'
l = s.split()
print(min(l, key=len))
```

```
I
```

[11]:
```
#Reverse the order of words in a string.
import re
s = 'This is a string to try'
z = re.split('\W+', s)
z.reverse()
z1=' '.join(z)
print(z1)
```

```
try to string a is This
```

[12]:
```
#Check if a string is alphanumeric.
string = "abc123"
print(string.isalnum())
```

```
True
```

[13]:
```
#Extract all digits from a string.
import re
s = '300 gm 200 kgm some more stuff a number: 439843'
print(re.findall('\d+', s))
```

```
['300', '200', '439843']
```

[14]:
```
#Extract all alphabets from a string.
import re
st="These 10 guesses are great"
word1 = " ".join(re.findall("[a-zA-Z]+", st))
print(word1)
```

```
These guesses are great
```

[15]:
```
#Count the number of uppercase letters in a string.
#Count the number of lowercase letters in a string.
Str="UnitedStatesOfAmerica"
lower=0
upper=0
for i in Str:
      if(i.islower()):
            lower+=1
      else:
            upper+=1
print("The number of lowercase characters is:",lower)
print("The number of uppercase characters is:",upper)
```

```
The number of lowercase characters is: 17
The number of uppercase characters is: 4
```

[16]:
```
#Swap the case of each character in a string.
str = "This is string example....wow!!!";
print(str.swapcase())
```

```
tHIS IS STRING EXAMPLE…WOW!!!
```

[17]:
```
#Remove a specific word from a string.
a1 = "remove word from this"
a2 = a1.replace("word", '')
print(a2)
```

```
remove  from this
```

[52]:
```
#Check if a string is a valid email address.
import re

# Define a function for
# for validating an Email
def check(s):
    pat = r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,7}\b'
    if re.match(pat,s):
        print("Valid Email")
    else:
        print("Invalid Email")

# Driver Code
if __name__ == '__main__':

    # Enter the email
    email = "ankitrai326@gmail.com"

    # calling run function
    check(email)

    email = "my.ownsite@our-earth.org"
    check(email)
    email = "ankitrai326.com"
    check(email)
```

```
Valid Email
Valid Email
Invalid Email
```

[27]:
```
#Extract the domain name from an email address string.

# initializing strings
test_str = 'vidya@majesco.com'

# printing original string
print("The original string is : " + test_str)

# slicing domain name using slicing
res = test_str[test_str.index('@') + 1 : ]

# printing result
print("The extracted domain name : " + res)
```

```
The original string is : vidya@majesco.com
The extracted domain name : majesco.com
```

[28]:
```
#Replace multiple spaces in a string with a single space.
mystring = 'Here is  some   text   I      wrote   '
mystring1=' '.join(mystring.split())
print(mystring1)
```

```
Here is some text I wrote
```

[29]:
```
#Extract the protocol (http or https) from a URL string.
given_url = 'https://www.google.com'
print(given_url.replace('https://',''))
```

```
www.google.com
```

[31]:
```
#Find the frequency of each character in a string.
test_str = "Smart people"

# using naive method to get count
# of each element in string
all_freq = {}

for i in test_str:
    if i in all_freq:
        all_freq[i] += 1
    else:
        all_freq[i] = 1

# printing result
print(all_freq)
```

```
{'S': 1, 'm': 1, 'a': 1, 'r': 1, 't': 1, ' ': 1, 'p': 2, 'e': 2, 'o': 1, 'l': 1}
```

[39]:
```
#Check if a string contains only digits.
def contains_only_digits(input_str):

    for char in input_str:
        if not char.isdigit():
            return False
    return True

my_string = "1234"
if contains_only_digits(my_string):
    print("The string contains only digits!")
else:
    print("The string does not contain only digits.")
```

```
The string contains only digits!
```

[40]:
```
#Check if a string contains only alphabets.
if 'hello'.isalpha():
    print("It's all letters")
```

```
It's all letters
```

[41]:
```
#Convert a string to a list of characters.
s = "Somesh Ramesh"
x = list(s)
print(x)
```

```
['S', 'o', 'm', 'e', 's', 'h', ' ', 'R', 'a', 'm', 'e', 's', 'h']
```

[42]:
```
#Check if two strings are anagrams.
str1 = "Race"
str2 = "Care"

# convert both the strings into lowercase
str1 = str1.lower()
str2 = str2.lower()

# check if length is same
if(len(str1) == len(str2)):

    # sort the strings
    sorted_str1 = sorted(str1)
    sorted_str2 = sorted(str2)

    # if sorted char arrays are same
    if(sorted_str1 == sorted_str2):
        print(str1 + " and " + str2 + " are anagram.")
    else:
        print(str1 + " and " + str2 + " are not anagram.")

else:
    print(str1 + " and " + str2 + " are not anagram.")
```

```
race and care are anagram.
```

[46]:
```
#Encode a string using a Caesar cipher.

def encrypt_text(plaintext,n):
    ans = ""
    # iterate over the given text
    for i in range(len(plaintext)):
        ch = plaintext[i]

        # check if space is there then simply add space
        if ch==" ":
            ans+=" "
        # check if a character is uppercase then encrypt it accordingly
        elif (ch.isupper()):
            ans += chr((ord(ch) + n-65) % 26 + 65)
        # check if a character is lowercase then encrypt it accordingly

        else:
            ans += chr((ord(ch) + n-97) % 26 + 97)

    return ans

plaintext = "HELLO EVERYONE"
n = 1
print("Plain Text is : " + plaintext)

print("Cipher Text is : " + encrypt_text(plaintext,n))
```

```
Plain Text is : HELLO EVERYONE
Cipher Text is : IFMMP FWFSZPOF
```

[47]:
```
#Check if a string contains any special characters.
def has_special_char(s):
    for c in s:
        if not (c.isalpha() or c.isdigit() or c == ' '):
            return True
    return False


# Test the function
s = "Hello World"
if has_special_char(s):
    print("The string contains special characters.")
else:
    print("The string does not contain special characters.")

s = "Hello@World"
if has_special_char(s):
    print("The string contains special characters.")
else:
    print("The string does not contain special characters.")
```

```
The string does not contain special characters.
The string contains special characters.
```

[48]:
```
#Split a string into a list of words.
lst =  "I am proud of my country"
print( lst.split())
```

```
['I', 'am', 'proud', 'of', 'my', 'country']
```

[50]:
```
#Find the longest word in a string.
s = 'I am not at all well'
l = s.split()
print(max(l, key=len))
```

```
well
```

[1]:
```
#Create a list with integers from 1 to 10.
list_of_numbers = list(range(1, 11))
print(list_of_numbers)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

[4]:
```
#Find the length of a list without using the `len()` function.
list_of_numbers = list(range(1, 11))
print(len(list_of_numbers))
```

```
10
```

[5]:
```
#Append an element to the end of a list.
my_list = ['element1', 'element2']
```

```
my_list.append('element3')
print(my_list)
```

```
['element1', 'element2', 'element3']
```

[6]:
```
#Insert an element at a specific index in a list.
myList = ['one', 'two', 'three']

myList.insert(0, 'zero')

print(myList)
```

```
['zero', 'one', 'two', 'three']
```

[7]:
```
#Remove an element from a list by its value.
#original list
programming_languages = ["JavaScript", "Python", "Java", "C++"]

#print original list
print(programming_languages)

# remove the value 'JavaScript' from the list
programming_languages.remove("JavaScript")

#print updated list
print(programming_languages)
```

```
['JavaScript', 'Python', 'Java', 'C++']
['Python', 'Java', 'C++']
```

[8]:
```
#Remove an element from a list by its index.
# input list
inputList = ["Welcome", "to", "tutorialspoint", "python"]

# Enter the index at which the list item is to be deleted
givenIndex = 3



# deleting the list item at the given index using the del keyword
del inputList[givenIndex]

# printing the list after deleting a specified list item
print("List after deleting specified list item:", inputList)
```

```
List after deleting specified list item: ['Welcome', 'to', 'tutorialspoint']
```

[9]:
```
#Check if an element exists in a list.
# List of string
listOfStrings = ['Hi' , 'hello', 'at', 'this', 'there', 'from']

# check if element exist in list using 'in'
if 'at' in listOfStrings :
    print("Yes, 'at' found in List : " , listOfStrings)
```

```
Yes, 'at' found in List :  ['Hi', 'hello', 'at', 'this', 'there', 'from']
```

[10]:
```
#Find the index of the first occurrence of an element in a list.
arr = [1, 3, 6, 2, 4, 6]
print ("The original array is: ", arr)
print()

specified_item = 6

# Get index of the first occurrence of the specified item
item_index = arr.index(specified_item)

print('The index of the first occurrence of the specified item is:',item_index)
```

```
The original array is:  [1, 3, 6, 2, 4, 6]

The index of the first occurrence of the specified item is: 2
```

[11]:
```
#Count the occurrences of an element in a list.
# Python code to count the number of occurrences
def countX(lst, x):
    count = 0
    for ele in lst:
        if (ele == x):
            count = count + 1
    return count


lst = [8, 6, 8, 10, 8, 20, 10, 8, 8]
x = 8
print('{} has occurred {} times'.format(x,
                                        countX(lst, x)))
```

```
8 has occurred 5 times
```

[12]:
```
#Reverse the order of elements in a list.
def Reverse(lst):
   new_lst = lst[::-1]
   return new_lst
```

```
lst = [10, 11, 12, 13, 14, 15]
print(Reverse(lst))
```

```
[15, 14, 13, 12, 11, 10]
```

[13]:
```
#Sort a list in ascending order.
# a list of numbers
my_numbers = [10, 8, 3, 22, 33, 7, 11, 100, 54]

#sort list in-place in ascending order
my_numbers.sort()

#print modified list
print(my_numbers)
```

```
[3, 7, 8, 10, 11, 22, 33, 54, 100]
```

[14]:
```
#Sort a list in descending order.
# a list of numbers
my_numbers = [10, 8, 3, 22, 33, 7, 11, 100, 54]

#sort list in-place in descending order
my_numbers.sort(reverse=True)

#print modified list
print(my_numbers)
```

```
[100, 54, 33, 22, 11, 10, 8, 7, 3]
```

[15]:
```
#Create a list of even numbers from 1 to 20.
# Python program to print Even Numbers in a List

# list of numbers
list1 = [1,2,3,4,5,6,7,8.9,10,11,12,13,14,15,16,17,18,19,20]

# iterating each number in list
for num in list1:

    # checking condition
    if num % 2 == 0:
        print(num, end=" ")
```

```
2 4 6 10 12 14 16 18 20
```

[16]:
```
#Create a list of odd numbers from 1 to 20.
# Python program to print odd Numbers in a List
# list of numbers
list1 = [1,2,3,4,5,6,7,8.9,10,11,12,13,14,15,16,17,18,19,20]

# iterating each number in list
for num in list1:

    # checking condition
    if num % 2 != 0:
       print(num, end=" ")
```

```
1 3 5 7 8.9 11 13 15 17 19
```

[17]:
```
#Find the sum of all elements in a list.
numbers = [1,2,3,4,5,1,4,5]

Sum = sum(numbers)
print(Sum)
```

```
25
```

[18]:
```
#Find the maximum value in a list.
heights = [100, 2, 300, 10, 11, 1000]
largest_number = heights[0]
for number in heights:
    if number > largest_number:
        largest_number = number
print(largest_number)
```

```
1000
```

[19]:
```
#Find the minimum value in a list.
heights = [100, 2, 300, 10, 11, 1000]
smallest_number = heights[0]
for number in heights:
    if number < smallest_number:
        smallest_number = number
print(smallest_number)
```

```
2
```

[20]:
```
#Create a list of squares of numbers from 1 to 10.
l = []

for i in range(1, 10):
    l.append(i * i)

print("List with square of integers from 1 to 50:")
print(l)
```

```
List with square of integers from 1 to 50:
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

[21]:
```
#Create a list of squares of numbers from 1 to 10.
l = []

for i in range(1, 10):
    l.append(i * i)

print("List with square of integers from 1 to 10:")
print(l)
```

```
List with square of integers from 1 to 10:
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

[26]:
```
#Create a list of random numbers.
import random
randomlist = []
for i in range(0,5):
 n = random.randint(1,30)
 randomlist.append(n)
 print(randomlist)
```

```
[15]
[15, 22]
[15, 22, 2]
[15, 22, 2, 8]
[15, 22, 2, 8, 3]
```

[27]:
```
#Remove duplicates from a list.
# initializing list
test_list = [1, 5, 3, 6, 3, 5, 6, 1]
print ("The original list is : "
        + str(test_list))

# using set() to remove duplicated from list
test_list = list(set(test_list))

# printing list after removal
# distorted ordering
print ("The list after removing duplicates : "
        + str(test_list))
```

```
The original list is : [1, 5, 3, 6, 3, 5, 6, 1]
The list after removing duplicates : [1, 3, 5, 6]
```

[28]:
```
#Find the common elements between two lists.
list1 = [1,2,3,4,5,6]
```

```
list2 = [3, 5, 7, 9]
print(list(set(list1).intersection(list2)))
```

```
[3, 5]
```

[33]:
```
li1 = [10, 15, 20, 25, 30, 35, 40]
li2 = [25, 40, 35]

temp3 = []
for element in li1:
    if element not in li2:
        temp3.append(element)

        print(temp3)
```

```
[10]
[10, 15]
[10, 15, 20]
[10, 15, 20, 30]
```

[35]:
```
#Merge two lists.
list1 = ['datagy', 'is', 'a', 'site']
list2 = ['to', 'learn', 'python']
list3 = list1 + list2
print(list3)
```

```
['datagy', 'is', 'a', 'site', 'to', 'learn', 'python']
```

[36]:
```
def multiplyList(myList):

    # Multiply elements one by one
    result = 1
    for x in myList:
        result = result * x
    return result


# Driver code
list1 = [1, 2, 3]
list2 = [3, 2, 4]
print(multiplyList(list1))
print(multiplyList(list2))
```

```
6
24
```

[37]:
```
#Multiply all elements in a list by 2.
def even(x):
    return x % 2 == 0

a = [2, 5, 7, 8, 10, 13, 16]

result = filter(even, a)
print('Original List :', a)
print('Filtered List :', list(result))
```

```
Original List : [2, 5, 7, 8, 10, 13, 16]
Filtered List : [2, 8, 10, 16]
```

[38]:
```
#Filter out all even numbers from a list.
list = ['5', '12', '4', '3', '5', '14', '16', '-2', '4', 'test']
list2 = []
for _ in list:
     try:
         list2.append(int(_))
     except:
         pass

print(list2)
```

```
[5, 12, 4, 3, 5, 14, 16, -2, 4]
```

[42]:
```
#Convert a list of strings to a list of integers.
lis = ['1', '-4', '3', '-6', '7']
res = [eval(i) for i in lis]
print("Modified list is: ", res)
```

```
Modified list is:  [1, -4, 3, -6, 7]
```

[43]:
```
#Convert a list of integers to a list of strings.
my_list = [[1], [2, 3], [4, 5, 6, 7]]

flat_list = [num for sublist in my_list for num in sublist]
print(flat_list)
```

```
[1, 2, 3, 4, 5, 6, 7]
```

[44]:
```
#Flatten a nested list.
my_list = [[1], [2, 3], [4, 5, 6, 7]]

flat_list = [num for sublist in my_list for num in sublist]
print(flat_list)
```

```
[1, 2, 3, 4, 5, 6, 7]
```

[1]:
```
#Check if a list is sorted.
test_list = [1, 4, 5, 8, 10]

# printing original list
print ("Original list : " + str(test_list))


flag = 0
i = 1
while i < len(test_list):
    if(test_list[i] < test_list[i - 1]):
        flag = 1
    i += 1

# printing result
if (not flag) :
    print ("Yes, List is sorted.")
else :
    print ("No, List is not sorted.")
```

```
Original list : [1, 4, 5, 8, 10]
Yes, List is sorted.
```

[2]:
```
#Rotate a list to the left by `n` positions.
# initializing list
test_list = [1, 4, 6, 7, 2]

# printing original list
print ("Original list : " + str(test_list))

# using slicing to left rotate by 3
test_list = test_list[3:] + test_list[:3]

# Printing list after left rotate
print ("List after left rotate by 3 : " + str(test_list))

# using slicing to right rotate by 3
# back to Original
test_list = test_list[-3:] + test_list[:-3]

# Printing after right rotate
print ("List after right rotate by 3(back to original) : "
                                        + str(test_list))
```

```
Original list : [1, 4, 6, 7, 2]
List after left rotate by 3 : [7, 2, 1, 4, 6]
List after right rotate by 3(back to original) : [1, 4, 6, 7, 2]
```

[3]:
```python
#Rotate a list to the right by `n` positions.
# Python program to right rotate a list by n

# Returns the rotated list

def rightRotate(lists, num):
    output_list = []

    # Will add values from n to the new list
    for item in range(len(lists) - num, len(lists)):
        output_list.append(lists[item])

    # Will add the values before
    # n to the end of new list
    for item in range(0, len(lists) - num):
        output_list.append(lists[item])

    return output_list


# Driver Code
rotate_num = 3
list_1 = [1, 2, 3, 4, 5, 6]

print(rightRotate(list_1, rotate_num))
```

```
[4, 5, 6, 1, 2, 3]
```

[4]:
```python
#Create a list of prime numbers up to 50.
def prime_numbers(n):
    primes = []
    for i in range(2, n + 1):
        for j in range(2, int(i ** 0.5) + 1):
            if i%j == 0:
                break
        else:
            primes.append(i)
    return primes

prime_list = prime_numbers(50)
print(prime_list)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

[5]:
```python
#Split a list into chunks of size `n`.

my_list = ['geeks', 'for', 'geeks', 'like',
            'geeky','nerdy', 'geek', 'love',
                'questions','words', 'life']

# Yield successive n-sized
# chunks from l.
def divide_chunks(l, n):

    # looping till length l
    for i in range(0, len(l), n):
        yield l[i:i + n]

# How many elements each
# list should have
n = 5

x = list(divide_chunks(my_list, n))
print (x)
```

```
[['geeks', 'for', 'geeks', 'like', 'geeky'], ['nerdy', 'geek', 'love',
'questions', 'words'], ['life']]
```

[6]:
```python
#Find the second largest number in a list.
# Python program to find second largest
# number in a list

# list of numbers - length of
# list should be at least 2
list1 = [10, 20, 4, 45, 99]

mx = max(list1[0], list1[1])
secondmax = min(list1[0], list1[1])
n = len(list1)
for i in range(2,n):
    if list1[i] > mx:
        secondmax = mx
        mx = list1[i]
    elif list1[i] > secondmax and \
        mx != list1[i]:
        secondmax = list1[i]
    elif mx == secondmax and \
        secondmax != list1[i]:
          secondmax = list1[i]

print("Second highest number is : ",\
      str(secondmax))
```

```
36.


37.

def convert(lst):
   res_dict = {}
   for i in range(0, len(lst), 2):
       res_dict[lst[i]] = lst[i + 1]
   return res_dict

lst = ['a', 1, 'b', 2, 'c', 3]
print(convert(lst))
```

```
Second highest number is :  45
{'a': 1, 'b': 2, 'c': 3}
```

[7]:
```
#Convert a list to a dictionary where list elements become keys and their
#indices become values.
37.

def convert(lst):
   res_dict = {}
   for i in range(0, len(lst), 2):
       res_dict[lst[i]] = lst[i + 1]
   return res_dict

lst = ['a', 1, 'b', 2, 'c', 3]
print(convert(lst))
```

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

[8]:
```
#Shuffle the elements of a list randomly.
import random
nums = [1, 2, 3, 4, 5]
print("Original list:")
print(nums)
random.shuffle(nums)
print("Shuffle list:")
print(nums)
```

```
Original list:
[1, 2, 3, 4, 5]
Shuffle list:
[1, 3, 4, 2, 5]
```

[9]:
```python
#Create a list of the first 10 factorial numbers.
# Python 3 program to find
# factorial of given number
def factorial(n):

    # single line to find factorial
    return 1 if (n==1 or n==0) else n * factorial(n - 1)

# Driver Code
num = 5
print("Factorial of",num,"is",factorial(num))
```

```
Factorial of 5 is 120
```

[23]:
```python
#Remove all elements from a list.
a = [1, 2, 3, 4, 5]

print(a)    # prints [1, 2, 3, 4, 5]
a.clear()
print(a)    # prints []
```

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

[24]:
```python
#Replace negative numbers in a list with 0.
list1 = [-10,1,2,3,9,-1]

list2 = [0 if i < 0 else i for i in list1]

print(list2)
```

```
[0, 1, 2, 3, 9, 0]
```

[25]:
```python
#Convert a string into a list of words.
import re
str1 = "Hello Everyone Welcome to Tutorialspoint"

print("The given string is")
print(str1)

print("The strings after the split are")
res = re.split('\s+', str1)
print(res)
```

```
The given string is
Hello Everyone Welcome to Tutorialspoint
The strings after the split are
['Hello', 'Everyone', 'Welcome', 'to', 'Tutorialspoint']
```

[26]:
```python
#Convert a list of words into a string.
import re
str1 = "Hello Everyone Welcome to Tutorialspoint"

print("The given string is")
print(str1)

print("The strings after the split are")
res = re.split('\s+', str1)
print(res)
```

```
The given string is
Hello Everyone Welcome to Tutorialspoint
The strings after the split are
['Hello', 'Everyone', 'Welcome', 'to', 'Tutorialspoint']
```

[27]:
```python
#. Create a list of the first `n` powers of 2.
def listToString(s):

    # initialize an empty string
    str1 = ""

    # traverse in the string
    for ele in s:
        str1 += ele

    # return string
    return str1
# Driver code
s = ['Geeks', 'for', 'Geeks']
print(listToString(s))
```

```
GeeksforGeeks
```

[28]:
```python
#Find the shortest string in a list of strings.
# initialize list
test_list = ['gfg', 'is', 'best', 'for', 'geeks']

# printing original list
print("The original list : " + str(test_list))

# Longest String in list
# using loop
max_len = -1
for ele in test_list:
    if len(ele) > max_len:
        max_len = len(ele)
        res = ele

# printing result
print("Maximum length string is : " + res)
```

```
The original list : ['gfg', 'is', 'best', 'for', 'geeks']
Maximum length string is : geeks
```

[29]:
```
#Create a list of the first `n` triangular numbers.
x = ['apple', 'banana', 'mango']
shortest = min(x, key=len)
print(shortest)
```

```
apple
```

[30]:
```
#Create a list of the first `n` triangular numbers.
def triangular_series( n ):
    j = 1
    k = 1

    # For each iteration increase j
    # by 1 and add it into k
    for i in range(1, n + 1):
        print(k, end = ' ')
        j = j + 1 # Increasing j by 1

        # Add value of j into k and update k
        k = k + j

n = 5
triangular_series(n)
```

```
1 3 6 10 15
```

[31]:
```
#Check if a list contains another list as a subsequence.
def check_list_contained(A, B):
  # convert list A to string
    A_str = ' '.join(map(str, A))
    # convert list B to string
    B_str = ' '.join(map(str, B))
    # find all instances of A within B
    instances = re.findall(A_str, B_str)

    # return True if any instances were found, False otherwise
    return len(instances) > 0
```

```
# Initializing lists
A = ['x', 'y', 'z']
B = ['x', 'a', 'y', 'x', 'b', 'z']

print(check_list_contained(A, B))
```

```
False
```

[32]:
```
#Swap two elements in a list by their indices.
def swapPositions(list, pos1, pos2):

    list[pos1], list[pos2] = list[pos2], list[pos1]
    return list

# Driver function
List = [23, 65, 19, 90]
pos1, pos2  = 1, 3

print(swapPositions(List, pos1-1, pos2-1))
```

```
[19, 65, 23, 90]
```

[3]:
```
#Create a tuple with integers from 1 to 5.
my_tuple = (1, 2, 3, 4, 5)
for val in my_tuple:
    print(val)
```

```
1
2
3
4
5
```

[6]:
```
#Access the third element of a tuple.
tuples = ('Spark','Python','Pandas','Pyspark','Java')
result = tuples[2]
print(result)
```

```
Pandas
```

[7]:
```
#Find the length of a tuple without using the `len()` function.
tuples = ('Spark','Python','Pandas','Pyspark','Java')

count = 0

for i in tuples:

    count+=1
```

```
print(count)
```

```
5
```

[14]:
```
#Count the occurrences of an element in a tuple.
tuples = ('Spark','Python','Pandas','Pyspark','Java','Spark')

count = 0


for val in tuples:
    if (val == 'Spark'):
       count+=1
print(count)
```

```
2
```

[15]:
```
#Find the index of the first occurrence of an element in a tuple.
test = ("Canada", "India", "Canada", "Japan", "Italy", "Canada")

idx = test.index("Japan")

print(idx)
```

```
3
```

[16]:
```
#Check if an element exists in a tuple.
test_tup = (10, 4, 5, 6, 8)

# printing original tuple
print("The original tuple : " + str(test_tup))

# initialize N
N = 6

# Check if element is present in tuple
# using loop
res = False
for ele in test_tup:
    if N == ele:
        res = True
        break

# printing result
print("Does tuple contain required value ? : " + str(res))
```

```
The original tuple : (10, 4, 5, 6, 8)
Does tuple contain required value ? : True
```

[20]:
```
#Convert a tuple to a list.
tuples = (0, 2, 4, 6, 8)
mylist = list(tuples)
print(mylist)
print(type(mylist))
```

```
[0, 2, 4, 6, 8]
<class 'list'>
```

[21]:
```
#Convert a list to a tuple.
def convert(list):
    return tuple(list)

# Driver function
list = [1, 2, 3, 4]
print(convert(list))
```

```
(1, 2, 3, 4)
```

[22]:
```
#Unpack the elements of a tuple into variables.
a = ("MNNIT Allahabad", 5000, "Engineering")

# this lines UNPACKS values
# of variable a
(college, student, type_ofcollege) = a

# print college name
print(college)

# print no of student
print(student)

# print type of college
print(type_ofcollege)
```

```
MNNIT Allahabad
5000
Engineering
```

[23]:
```
#Create a tuple of even numbers from 1 to 10.
evTuple = (1, 2, 3, 4, 5, 6, 7, 8,9,10)
print("Tuple Items = ", evTuple)

print("\nThe Even Numbers in this evTuple Tuple are:")
for i in range(len(evTuple)):
    if(evTuple[i] % 2 == 0):
        print(evTuple[i], end = "  ")
```

```
Tuple Items =  (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

The Even Numbers in this evTuple Tuple are:
2  4  6  8  10
```

[24]:
```
#Create a tuple of odd numbers from 1 to 10.
evTuple = (1, 2, 3, 4, 5, 6, 7, 8,9,10)
print("Tuple Items = ", evTuple)

print("\nThe Even Numbers in this evTuple Tuple are:")
for i in range(len(evTuple)):
    if(evTuple[i] % 2 != 0):
        print(evTuple[i], end = "  ")
```

```
Tuple Items =  (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

The Even Numbers in this evTuple Tuple are:
1  3  5  7  9
```

[25]:
```
#Concatenate two tuples.
test_tup1 = (1, 3, 5)
test_tup2 = (4, 6)

# printing original tuples
print("The original tuple 1 : " + str(test_tup1))
print("The original tuple 2 : " + str(test_tup2))

# Ways to concatenate tuples
# using + operator
res = test_tup1 + test_tup2

# printing result
print("The tuple after concatenation is : " + str(res))
```

```
The original tuple 1 : (1, 3, 5)
The original tuple 2 : (4, 6)
The tuple after concatenation is : (1, 3, 5, 4, 6)
```

[26]:
```
#Repeat a tuple three times.
# initialize tuple
test_tup = (1, 3)

# printing original tuple
print("The original tuple : " + str(test_tup))

# initialize N
N = 4
```

```
# Repeating tuples N times
# using * operator
res = ((test_tup, ) * N)

# printing result
print("The duplicated tuple elements are : " + str(res))
```

```
The original tuple : (1, 3)
The duplicated tuple elements are : ((1, 3), (1, 3), (1, 3), (1, 3))
```

[27]:
```
#Check if a tuple is empty.
Mytuple=()
# Using not operator
if not Mytuple:
    print ("Mytuple is empty")
else:
    print ("Mytuple is not empty")
# Printing the tuple
print(Mytuple)
```

```
Mytuple is empty
()
```

[28]:
```
#Create a nested tuple.
# initialize tuples
test_tup1 = (3, 4),
test_tup2 = (5, 6),

# printing original tuples
print("The original tuple 1 : " + str(test_tup1))
print("The original tuple 2 : " + str(test_tup2))

# Concatenating tuples to nested tuples
# using + operator + ", " operator during initialization
res = test_tup1 + test_tup2

# printing result
print("Tuples after Concatenating : " + str(res))
```

```
The original tuple 1 : ((3, 4),)
The original tuple 2 : ((5, 6),)
Tuples after Concatenating : ((3, 4), (5, 6))
```

[30]:
```
#Access the first element of a nested tuple.
data = [(1, 'sravan'), (2, 'ojaswi'),
        (3, 'bobby'), (4, 'rohith'),
        (5, 'gnanesh')]
```

```
# iterate using for loop
# to access first elements
for i in data:
    print(i[0])
```

```
1
2
3
4
5
```

[31]:
```
# Creating a tuple having one element
var2 = ("hello",)
print(type(var2))  # <class 'tuple'>
```

```
<class 'tuple'>
```

[33]:
```
#Compare two tuples.
# Using the != operator
tuple1 = (2, 4, 6)
tuple2 = (2, 4, 6)
tuple3 = (1, 3, 5)
result = tuple1 != tuple2
print(result)
```

```
False
```

[34]:
```
#Delete a tuple.
tup=('tutorials', 'point', 2022,True)
print(tup)
del(tup)
print("After deleting the tuple:")
print(tup)
```

```
('tutorials', 'point', 2022, True)
After deleting the tuple:
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[34], line 6
      4 del(tup)
      5 print("After deleting the tuple:")
----> 6 print(tup)

NameError: name 'tup' is not defined
```

[35]:
```
#Slice a tuple.
tuple= ('a','b','c','d','e','f','g','h','i','j')
print(tuple[0:6])
print(tuple[1:9:2])
print(tuple[-1:-5:-2])
```

```
('a', 'b', 'c', 'd', 'e', 'f')
('b', 'd', 'f', 'h')
('j', 'h')
```

[1]:
```
#Find the maximum value in a tuple.
aTuple = (2, 5, 8, 1, 4, 3)
result = max(aTuple)
print('Maximum :', result)
```

```
Maximum : 8
```

[2]:
```
#Find the minimum value in a tuple.
aTuple = (2, 5, 8, 1, 4, 3)
result = min(aTuple)
print('Maximum :', result)
```

```
Maximum : 1
```

[15]:
```
#Convert a string to a tuple of characters.
my_str_1 = "a, b, c, d, e, f, g, h, i"

print ("The string is : " )
print(my_str_1)

my_result = tuple(map(str, my_str_1.split(', ')))

print("The tuple after converting from a string is : ")
print(my_result)
```

```
The string is :
a, b, c, d, e, f, g, h, i
The tuple after converting from a string is :
('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i')
```

[11]:
```
#Convert a tuple of characters to a string.
tuples = ('a', 'b', 'c', 'd')

# Use str.join() function
# to convert a tuple to a string
string = " ".join(tuples)
print(string)
```

```
a b c d
```

[5]:
```python
#Create a tuple with different data types
tuplex = ("tuple", False, 3.2, 1)
print(tuplex)
```

```
('tuple', False, 3.2, 1)
```

[17]:
```python
#Check if two tuples are identical.
tuple1 = (1, 2, 3)
tuple2 = (1, 2, 4)
tuple3 = (1, 2, 3)

print(tuple1 == tuple2)
print(tuple1 == tuple3)
```

```
False
True
```

[18]:
```python
#Sort the elements of a tuple.
aTuple = (2, 5, 8, 1, 9, 3, 7)
result = sorted(aTuple)
result = tuple(result)
print('Sorted Tuple :', result)
```

```
Sorted Tuple : (1, 2, 3, 5, 7, 8, 9)
```

[19]:
```python
#Convert a tuple of integers to a tuple of strings.
def tuple_int_str(tuple_str):
    result = tuple((str(x[0]), str(x[1])) for x in tuple_str)
    return result

tuple_str =  ((333, 33), (1416, 55))
print("Original tuple values:")
print(tuple_str)
print("\nNew tuple values:")
print(tuple_int_str(tuple_str))
```

```
Original tuple values:
((333, 33), (1416, 55))

New tuple values:
(('333', '33'), ('1416', '55'))
```

[20]:
```python
def tuple_int_str(tuple_str):
    result = tuple((int(x[0]), int(x[1])) for x in tuple_str)
    return result
```

```python
tuple_str =  (('333', '33'), ('1416', '55'))
print("Original tuple values:")
print(tuple_str)
print("\nNew tuple values:")
print(tuple_int_str(tuple_str))
```

```
Original tuple values:
(('333', '33'), ('1416', '55'))

New tuple values:
((333, 33), (1416, 55))
```

[21]:
```python
#Merge two tuples.
test_tup1 = (1, 3, 5)
test_tup2 = (4, 6)

# printing original tuples
print("The original tuple 1 : " + str(test_tup1))
print("The original tuple 2 : " + str(test_tup2))

# Ways to concatenate tuples
# using + operator
res = test_tup1 + test_tup2

# printing result
print("The tuple after concatenation is : " + str(res))
```

```
The original tuple 1 : (1, 3, 5)
The original tuple 2 : (4, 6)
The tuple after concatenation is : (1, 3, 5, 4, 6)
```

[22]:
```python
#Flatten a nested tuple.
ls = [('a','b','c'),('d','e','f'),('g','h','i')]

# iterate through list of tuples in a nested loop
flat_ls = []
for tup in ls:
    for item in tup:
        flat_ls.append(item)
# display the lists
print("Original list:", ls)
print("Flattened list:", flat_ls)
```

```
Original list: [('a', 'b', 'c'), ('d', 'e', 'f'), ('g', 'h', 'i')]
Flattened list: ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
```

[24]:
```python
#Create a tuple of the first 5 prime numbers.
import math

# Function to generate first n primes
def generatePrime(n):
    X = 0
    i = 2
    flag = False
    while(X < n):
        flag = True
        for j in range(2, math.floor(math.sqrt(i)) + 1):
            if (i%j == 0):
                flag = False
                break
        if(flag):
            print(i, end=" ")
            X+=1
        i+=1
    print()



# Test Case 1
N = 5

# Function call
generatePrime(N)
```

```
2 3 5 7 11
```

[25]:
```python
#Filter out all even numbers from a tuple.
# initializing list
test_list = [(6, 4, 2, 8), (5, 6, 7, 6), (8, 0, 2), (7, )]

# printing original list
print("The original list is : " + str(test_list))

# define function to check if all elements of a tuple are even
def all_even(t):
    return all(i % 2 != 0 for i in t)

# use map() and all() to filter tuples with all even elements
res_list = [t for t in test_list if all(map(all_even, [t]))]

# print results
print("Filtered Tuples : " + str(res_list))
```

```
The original list is : [(6, 4, 2, 8), (5, 6, 7, 6), (8, 0, 2), (7,)]
Filtered Tuples : [(7,)]
```

[26]:
```
#Multiply all elements in a tuple by 2.
my_tuple = (5, 3)

by_five = tuple(2 * elem for elem in my_tuple)

print(by_five)
```

```
(10, 6)
```

[29]:
```
#Create a tuple of random numbers.
import random
nums = []
for _ in range(10):
    nums.append(random.random())
nums = tuple(nums)
print(nums)
```

```
(0.2646474741993644, 0.3956435470140447, 0.2342810793806709, 0.62010860158961,
0.9533072261762859, 0.38472660026064653, 0.26845420252244967,
0.04480352132444254, 0.77981285308877, 0.8689401810608469)
```

[30]:
```
#Check if a tuple is sorted.
def is_tuple_sorted(t):
    for i in range(1, len(t)):
        # return False if the element is smaller than the previous element
        if t[i] < t[i-1]:
            return False
    return True
# create a tuple
t = (1, 2, 3, 4, 5)
# check if tuple is sorted
print(is_tuple_sorted(t))
```

```
True
```

[32]:
```
#Create a tuple from user input.
x = input('Enter the tuple : ')
x = tuple(int(a) for a in x.split(","))
print(x)
```

```
Enter the tuple :  1,2,3,4,5

(1, 2, 3, 4, 5)
```

[33]:
```
#Swap two elements in a tuple.
# create a tuple
t = ('Jim', 'Ben')
# convert the tuple to list
ls = list(t)
# swap the elements in the list using their index
ls[0], ls[1] = ls[1], ls[0]
# create a new tuple from the list elements
new_t = tuple(ls)
# print the new tuple
print(new_t)
```

```
('Ben', 'Jim')
```

[34]:
```
#Reverse the elements of a tuple.
def Reverse(tuples):
    new_tup = tuples[::-1]
    return new_tup

# Driver Code
tuples = ('z','a','d','f','g','e','e','k')
print(Reverse(tuples))
```

```
('k', 'e', 'e', 'g', 'f', 'd', 'a', 'z')
```

[36]:
```
#Find the longest string in a tuple of strings.
test_tuple= ('gfg', 'is', 'best', 'for', 'geeks')

# printing original tuple
print("The original tuple : " + str(test_list))

# Longest String in tuple
# using loop
max_len = -1
for ele in test_tuple:
    if len(ele) > max_len:
        max_len = len(ele)
        res = ele

# printing result
print("Maximum length string is : " + res)
```

```
The original tuple : ('gfg', 'is', 'best', 'for', 'geeks')
Maximum length string is : geeks
```

[38]:
```
#Find the shortest string in a tuple of strings.
test_tuple = ['gfg', 'is', 'best']
```

```
# printing original list
print("The original tuple : " + str(test_tuple))

# Minimum String length
# using min() + generator expression
res = min(len(ele) for ele in test_tuple)

# printing result
print("Length of minimum string is : " + str(res))
```

```
The original tuple : ['gfg', 'is', 'best']
Length of minimum string is : 2
```

[41]:
```
#Create a tuple of the first `n` triangular numbers.
from itertools import accumulate
limit = 10
tuple(accumulate(range(1, limit+1)))
```

[41]:
```
(1, 3, 6, 10, 15, 21, 28, 36, 45, 55)
```

[43]:
```
#Create a tuple of alternating 1s and 0s of length `n`.
count_1 = 1

# count of 0
count_0 = 1

# total length of tuple
size = 14

# initializing tuple cyclically
# using tuple comprehension
test_list = [1 if i % (count_1 + count_0) < count_1
             else 0 for i in range(size)]

# printing list after change
print("The list after initializing : " + str(test_list))
```

```
The list after initializing : [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
```

[1]:
```
#Add an element to a set.
nameSet = {"John", "Jane", "Doe"}

nameSet.add("Ihechikara")

print(nameSet)
```

```
{'Ihechikara', 'John', 'Jane', 'Doe'}
```

[2]:
```
#Discard element in a set
nameSet = {"John", "Jane", "Doe"}

nameSet.discard("John")

print(nameSet)
```

```
{'Jane', 'Doe'}
```

[6]:
```
   #Check if an element exists in a set.
   my_set = {1, 2, 3, 4, 5}
   if 6 not in my_set:
      print("6 is not present in the set")
   else:
      print("6 is present in the set")
```

```
6 is not present in the set
```

[8]:
```
#Find the length of a set without using the `len()` function.
inp_set = {1, 2, 3, 4, 5}
size = 0
for x in inp_set:
    size += 1
print(size)
```

```
5
```

[1]:
```
#Clear all elements from a set.
fruits = {"apple", "banana", "cherry"}

fruits.clear()

print(fruits)
```

```
set()
```

[2]:
```
#Create a set of even numbers from 1 to 10.
evens = {x for x in range(2, 11, 2)}
print(evens)
```

```
{2, 4, 6, 8, 10}
```

[4]:
```
#Create a set of odd numbers from 1 to 10.
odds = {x for x in range(1, 10, 2)}
print(odds)
```

```
{1, 3, 5, 7, 9}
```

[5]:
```
#Find the union of two sets.
A = {2, 4, 5, 6}
B = {4, 6, 7, 8}

print("A U B:", A.union(B))
```

```
A U B: {2, 4, 5, 6, 7, 8}
```

[9]:
```
#Find the intersection of two sets.
A = {2, 4, 5, 6}
B = {4, 6, 7, 8}

print("Intersection is",A.intersection(B))
```

```
Intersection is {4, 6}
```

[11]:
```
#Find the difference between two sets.
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
diff1 = set1.difference(set2)
diff2 = set2.difference(set1)
print(diff1)
print(diff2)
```

```
{1, 2, 3}
{8, 6, 7}
```

[12]:
```
#Check if a set is a subset of another set.
A = {1, 2, 3}
B = {1, 2, 3, 4, 5}

# all items of A are present in B
print(A.issubset(B))
```

```
True
```

[13]:
```
A = {4, 1, 3, 5}
B = {6, 0, 4, 1, 5, 0, 3, 5}

print("A.issuperset(B) : ", A.issuperset(B))
print("B.issuperset(A) : ", B.issuperset(A))
```

```
A.issuperset(B) :  False
B.issuperset(A) :  True
```

[14]:
```
#Create a set from a list.
my_list = [1 ,2 ,3, 3, 4, 5, 5]
```

```
my_set = set()
# Using for loop
for num in my_list:
   my_set.add(num)
print(my_set)
```

```
{1, 2, 3, 4, 5}
```

[16]:
```
#Convert a set to a list.
myset={12,32,6,"sparkby","examples"}

# Convert to list
done=[]
for i in myset:
  done.append(i)
print(done)
```

```
[32, 'examples', 6, 12, 'sparkby']
```

[17]:
```
#Remove a random element from a set.
A = {2, 3, 7, 8, 45, 76}

print ('Popped:', A.pop()) # removes a random element
print ('Set:', A)
```

```
Popped: 2
Set: {3, 7, 8, 76, 45}
```

[18]:
```
#Pop an element from a set.
s1 = {9, 1, 0}
s1.pop()
print(s1)
```

```
{9, 1}
```

[20]:
```
#Check if two sets have no elements in common.
a = {23,45,78,8,56}
b = {42,55,26,87}
z = {87,46}
print("a :",a)
print("b :",b)
print("Z :",z)

print("\nCompare A and B : ",a.isdisjoint(b))
print("Compare B and Z : ",b.isdisjoint(z))
print("Compare A and Z : ",z.isdisjoint(a))
```

```
a : {23, 8, 56, 45, 78}
b : {42, 26, 55, 87}
Z : {46, 87}

Compare A and B :  True
Compare B and Z :  False
Compare A and Z :  True
```

[21]:
```python
#Find the symmetric difference between two sets.
A = {'a','b','c','d'}
B = {'a','f','g'}

symmetry = A.symmetric_difference(B)
print('The symmetry between A and B is=', symmetry)
```

```
The symmetry between A and B is= {'d', 'b', 'g', 'c', 'f'}
```

[22]:
```python
#Update a set with elements from another set.
myset = {12,34,56,3,45,67,89,1,6}
print("Actual set:",myset)

# Use update() method to add elements from a list
myset.update(["Hello","welcome"])
print("Set after adding list of elements:",myset)
```

```
Actual set: {1, 34, 67, 3, 6, 12, 45, 56, 89}
Set after adding list of elements: {1, 34, 67, 3, 6, 12, 45, 'Hello', 'welcome',
56, 89}
```

[24]:
```python
#Check if two sets are identical.
myset1 = {12,90,43,56}
myset2 = {43,56,12,90}

print("Set1 and Set2 equal? ",myset1 == myset2)
```

```
Set1 and Set2 equal?  True
```

[28]:
```python
#Create a frozen set.
letters = ('m', 'r', 'o', 't', 's')

fSet = frozenset(letters)
print('Frozen set is:', fSet)
```

```
Frozen set is: frozenset({'t', 'm', 'o', 'r', 's'})
```

[33]:
```python
#Create a set of squares of numbers from 1 to 5.

myset1 = {1,2,3,4,5}
```

```
for num in myset1:

   print(num**2)
```

```
1
4
9
16
25
```

[47]:
```
#Filter out all even numbers from a set.
myset1 = {1,2,3,4,5}

# Output Set initialisation
out = set()

for num in myset1:

    # checking condition
    if num % 2 == 0:
        out.add( num )


# printing output
print(out)
```

```
{2, 4}
```

[37]:
```
#Multiply all elements in a set by 2.

myset1 = {1,2,3,4,5}

for num in myset1:

   print(num*2)
```

```
2
4
6
8
10
```

[49]:
```
#Create a set of random numbers.
import random
```

```
Start = 9
Stop = 99
limit = 10

RandomSetOfIntegers = {random.randint(Start, Stop) for iter in range(limit)}
print(RandomSetOfIntegers)
```

```
{96, 99, 78, 49, 82, 83, 52, 85, 53, 93}
```

[1]:
```
#Check if a set is empty.
MySet = {}
# Using not operator
if not MySet:
    print ("set is empty")
else:
    print ("set is not empty")
```

```
set is empty
```

[3]:
```
#Create a nested set (hint: use frozenset).
xx = set([])
# Nested sets must be frozen
elements = frozenset([2,3,4])
xx.add(elements)
print(xx);
```

```
{frozenset({2, 3, 4})}
```

[4]:
```
#Remove an element from a set using the discard method.
def Remove(sets):
    sets.discard(20)
    print (sets)


sets = set([10, 20, 26, 41, 54, 20])
Remove(sets)
```

```
{41, 10, 54, 26}
```

[5]:
```
#Compare two sets.
# Create three sets
myset1 = {12,90,43,56}
myset2 = {43,56,12,90}
myset3 = {43,56,12}
print("Set 1: ",myset1)
print("Set 2: ",myset2)
print("Set 3: ",myset3)
```

```
# Check sets equality using == operator
print("Set1 and Set2 equal? ",myset1 == myset2)
print("Set1 and Set3 equal? ",myset1 == myset3)
```

```
Set 1:  {56, 90, 43, 12}
Set 2:  {56, 90, 43, 12}
Set 3:  {56, 43, 12}
Set1 and Set2 equal?  True
Set1 and Set3 equal?  False
```

[6]:
```
#Create a set from a string.
myStr = "pythonforbeginners"
mySet = set(myStr)
print("The input string is:", myStr)
print("The output set is:", mySet)
```

```
The input string is: pythonforbeginners
The output set is: {'h', 'o', 'b', 'r', 't', 'y', 'i', 'p', 'f', 's', 'e', 'g',
'n'}
```

[7]:
```
#Convert a set of strings to a set of integers.
set1= { '1', '2' }
set2 = set(map(int, set1))
print(set2)
```

```
{1, 2}
```

[8]:
```
#Convert a set of integers to a set of strings.
set1= { 1, 2 }
set2 = set(map(str, set1))
print(set2)
```

```
{'1', '2'}
```

[ ]:
```
#Find the maximum value in a set.
def MAX(sets):
    return (max(sets))

# Driver Code
sets = set([8, 16, 24, 1, 25, 3, 10, 65, 55])
print(MAX(sets))
```

[7]:
```
#Create a set from a tuple.
#   program to convert set to tuple
# create set
s = {'a', 'b', 'c', 'd', 'e'}

# print set
```

```
print(type(s), " ", s)

# call tuple() method
# this method convert set to tuple
t = tuple(s)

# print tuple
print(type(t), " ", t)
```

```
<class 'set'>   {'d', 'a', 'c', 'b', 'e'}
<class 'tuple'>   ('d', 'a', 'c', 'b', 'e')
```

[8]:
```
#Convert a set to a tuple.
#take a set of elements
mySet = {'apple', 'banana', 'cherry'}
#unpack set items and form tuple
output = (*mySet,)
print(f'Tuple : {output}')
```

```
Tuple : ('cherry', 'apple', 'banana')
```

[1]:
```
#Find the minimum value in a set.
def MIN(sets):
    return (max(sets))

sets = set([8, 16, 24, 1, 25, 3, 10, 65, 55])
print(MIN(sets))
```

```
65
```

[3]:
```
#Create a set from user input.
user_input = input('Enter space-separated integers: ')

my_set = set(int(item) for item in user_input.split())

print(my_set)
```

```
Enter space-separated integers:  1 2 3 4 5

{1, 2, 3, 4, 5}
```

[12]:
```
#Check if the intersection of two sets is empty.
L = [1,2,3,4,5,6]
M = [8,9,10]
if set(L) & set(M):
    print("intersection");
else:
   print("not a intersection")
```

```
not a intersection
```

[13]:
```
#Remove duplicates from a list using sets.
# initializing list
test_list = [1, 5, 3, 6, 3, 5, 6, 1]
print ("The original list is : "
        + str(test_list))

# using set() to remove duplicated from list
test_list = list(set(test_list))

# printing list after removal
# distorted ordering
print ("The list after removing duplicates : "
        + str(test_list))
```

```
The original list is : [1, 5, 3, 6, 3, 5, 6, 1]
The list after removing duplicates : [1, 3, 5, 6]
```

[14]:
```
#Check if two sets have the same elements, regardless of their count.
# Create three sets
myset1 = {12,90,43,56}

myset3 = {43,56,12}
print("Set 1: ",myset1)

print("Set 3: ",myset3)

# Check sets equality using == operator
print("Set1 and Set3 equal? ",myset1 == myset3)
```

```
Set 1:  {56, 90, 43, 12}
Set 3:  {56, 43, 12}
Set1 and Set3 equal?  False
```

[4]:
```
#Create a set of the first `n` powers of 2.
def squares(n):
    power = n
    square_set = set()
    for i in range(1,n+1):
        square_set.add(2 ** i)
    return square_set

print(squares(4))
```

```
{8, 16, 2, 4}
```

[5]:
```
#Find the common elements between a set and a list.

import numpy as np
def common_member(a, b):
    return list(np.intersect1d(a, b))

# Example usage:
a = {1, 2, 3, 4, 5}
b = {1, 2, 3, 4, 5}
common_elements = common_member(a, b)
print(common_elements)
```

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

[6]:
```
#Check if a set contains another set as a subset.
setA = {1, 2, 3, 4, 5}
setB = {1, 2, 3}
setC = {1, 2, 3, 6, 7}
print("setA: ", setA)
print("setB: ", setB)
print("setC: ", setC)

print("Is setB a subset of setA?: ", setB.issubset(setA))
print("Is setA a subset of setB?: ", setA.issubset(setB))
print("Is setC a subset of setA?: ", setC.issubset(setA))
```

```
setA:  {1, 2, 3, 4, 5}
setB:  {1, 2, 3}
setC:  {1, 2, 3, 6, 7}
Is setB a subset of setA?:  True
Is setA a subset of setB?:  False
Is setC a subset of setA?:  False
```

[10]:
```
#Create a set of alternating 1s and 0s of length `n`.
# count of 1
count_1 = 4

# count of 0
count_0 = 3

# total length of Set
size = 14

# initializing Set cyclically
# using list comprehension
test_Set = [1 if i % (count_1 + count_0) < count_1
             else 0 for i in range(size)]
```

```python
# printing list after change
print("The list after initializing : " + str(test_Set))
```

```
The list after initializing : [1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0]
```

[12]:
```python
#Merge multiple sets into one.
myset1 = {"one","two","three"}
myset2 = {"four","five","six"}
myset = myset1.union(myset2)
print(myset)
```

```
{'one', 'three', 'four', 'five', 'two', 'six'}
```

[ ]:
```python

```