20th august 2023 python basics

January 23, 2024

[1]:
```
# 1.reverse of string
string1 = input("enter a string:")
print(string1[::-1])
```

```
enter a string:vijay
yajiv
```

[2]:
```
# 2.check palindrome
n = input("enter a string:")
if n == n[::-1]:
    print("given string is palindrome", n )
else:
    print("given string is not a palindrome")
```

```
enter a string:vijiv
given string is palindrome vijiv
```

[3]:
```
# 3.convert string to upper case
string1 = input("enter string:")
string1.upper()
```

```
enter string:vijay deepu
```

[3]:
```
'VIJAY DEEPU'
```

[4]:
```
#4.covert string to lower
string= input("enter string:")
string.lower()
```

```
enter string:deep
```

[4]:
```
'deep'
```

[19]:
```
#5.count the number of vowels in a string
string1 = str(input("enter a string :"))
vowels=0
for i in string1:
    if  i in "a,e,i,o,u,A,E,I,O,U":
        vowels = vowels+1
print("the no of vowels are", vowels)
```

```
enter a string :vijay
the no of vowels are 2
```

[18]:
```
# 6.count the number of consonants in the string
string1 = str(input("enter a string :"))
consonants=0
for i in string1:
    if i not in "a,e,i,o,u,A,E,I,O,U":
        consonants= consonants+1
print("the no of consonants", consonants)
```

```
enter a string :vijay
the no of consonants 3
```

[2]:
```
# 7.remove all white spaces from a string
string1 = str(input("enter a string :"))
string2 = string1.replace(" ","")

print(string2)
```

```
enter a string :v    h     i
vhi
```

[ ]:
```
# 8.to find length of the string without using len()function
str1 =input("enter a string :")
count = 0
for i in str1:
    count = count+1
print("the length of the string", count)
```

[2]:
```
# 9.check if a string contains a specific word
str1= "this is python programming"
word = input("enter word")
if word in str1:
    print(True)
else:
    print(False)
```

```
enter wordThis
False
```

[8]:
```
#10. replace a word in string with another word
str1= "this is python programming"
str1.replace('programming','program')
```

[8]:
```
'this is python program'
```

[14]:
```
# 11.count the ocurrence of words in string
string2 = "hi today is tuesday and its already evening is"
string2.count('is')
```

[14]: 2

[4]:
```
# 12.find the first occurrence of a word in string
string2 = "hi today is tuesday and its already evening is"
first_word = string2.find("is")
print("the first ocurrence of the word is at",first_word)
```

```
the first ocurrence of the word is at 9
```

[7]:
```
# 13.find the last occurrence of a word in string
string3 = "He raced to the grocery store. He went inside but realized he forgot␣
 ↪his wallet. He raced back home to grab it. Once he found it, he raced to the␣
 ↪car again and drove back to the grocery store. "
word = string3.rfind("store")
print("the first ocurrence of the word is at",word)
```

```
the first ocurrence of the word is at 186
```

[8]:
```
# 14.split a string into a list of words
string4="hi i completed MCA"
string4.split()
```

[8]: ['hi', 'i', 'completed', 'MCA']

[25]:
```
# 15.join a list of words into a string
l1=['hi', 'i', 'completed', 'MCA']
' '.join(l1)
```

[25]: 'hi i completed MCA'

[22]:
```
# 16.Convert a string where words are separated by spaces to one where words␣
 ↪are separated by underscores
string5="xpulse is an adventure bike"
string5.replace(' ','_')
```

[22]: 'xpulse_is_an_adventure_bike'

[26]:
```
# 17. Check if a string starts with a specific word or phrase
string5="xpulse is an adventure bike"
string5.startswith('xpulse')
```

[26]: True

[30]:
```
# 18. Check if a string ends with a specific word or phrase
string5="xpulse is an adventure bike"
```

```
string5.endswith('bike')
```

[30]: True

[31]:
```
#19.Convert a string to title case (e.g., "hello world" to "Hello World")
string6 ="deep is a professional"
string6.title()
```

[31]: 'Deep Is A Professional'

[52]:
```
# 20. Find the longest word in a string
string6 ="deep is a professional"
s=string6.split()
max=0
for ele in s:
    if len(ele) > max:
        max=len(ele)
        max_word= ele
print("word with max elements",max_word)
print("length of max word is ",len(max_word))
```

```
word with max elements professional
length of max word is  12
```

[ ]:

[73]:
```
# 21. Find the shortest word in a string
s =input("enter string : ")
str=s.split()
min=len(str[0])
min_word=""
for ele in str:
    if len(ele)<min:
        min=len(ele)
        min_word=ele

print("smallest word in given string is ",min_word,"with lenth",len(min_word))
```

```
enter string : vijay and deep
smallest word in given string is  and with lenth 3
```

[2]:
```
# 22. Reverse the order of words in a string.
str = input("enter string : ")
words = str.split(' ')
out = ' '.join(reversed(words))
print(out)
```

```
enter string : deep vijay
vijay deep
```

[4]:
```
# 23.Check if a string is alphanumeric.
string1 = input("enter a string")
string1.isalnum()
```

```
enter a stringhello123
```

[4]:
```
True
```

[10]:
```
#24.Extract all digits from a string.
def extract_digits(text):
    digits = ""

    for char in text:
        if char.isdigit():
            digits += char

    return digits
```

[13]:
```
extract_digits("helloworld123hi4456789")
```

[13]:
```
'1234456789'
```

[15]:
```
# 25.Extract all alphabets from a string.
def extract_alphabets(text):
    alphabets = ""

    for char in text:
        if char.isalpha():
            alphabets += char

    return alphabets
```

[16]:
```
extract_alphabets("hello123hi234bye2334")
```

[16]:
```
'hellohibye'
```

[6]:
```
#26.Count the number of uppercase letters in a string.
def count_uppercase(string):
    u=0
    for i in string:
        if i.isupper():
            u =u+1

    return u
```

[8]:
```
count_uppercase("HIGH")
```

[8]: 4

[2]:
```
# 27.Count the number of lowercase letters in a string.

def count_lowercase(string):
    l=0
    for i in string:
        if i.islower():
            l=l+ 1

    return l
```

[3]:
```
count_lowercase("vijaya rama raju")
```

[3]: 14

[9]:
```
#28.Swap the case of each character in a string.
string1 = input("enter a string")
string1.swapcase()
```

```
enter a stringvijay
```

[9]: 'VIJAY'

[11]:
```
#29. Remove a specific word from a string.
def remove_word(input_string, word_to_remove):
    words = input_string.split()
    filtered_words = [word for word in words if word != word_to_remove]
    modified_string = ' '.join(filtered_words)
    return modified_string
```

[12]:
```
remove_word("hi this is vijay","this")
```

[12]: 'hi is vijay'

[25]:
```
#30.Check if a string is a valid email address.
import re
def is_valid_email(email):
    pattern = r'^[\w\.-]+@[\w\.-]+\.\w+$'
    if re.match(pattern, email):
        return True
    else:
        return False
```

[26]:
```
is_valid_email("vijayveera786@gmail.com")
```

[26]: True

[30]:
```
#31.Extract the username from an email address string.
email = input("enter your email")
username,mail=email.split('@')
print(username)
```

```
enter your emailvijayveera786@gmail.com
vijayveera786
```

[31]:
```
#32.Extract the username from an email address string.
email = input("enter your email")
mail,domainname=email.split('gmail')
print(domainname)
```

```
enter your emailvjjayveera@gmail.com
.com
```

[29]:
```
#33.Replace multiple spaces in a string with a single space.
string5 = ("vijay   rama    raju")
string5.replace("   "," ")
```

[29]:
```
'vijay rama  raju'
```

[32]:
```
#34. Check if a string is a valid URL.
from urllib.parse import urlparse

def is_valid_url(url):
    try:
        result = urlparse(url)
        return all([result.scheme, result.netloc])  # Check if both scheme and␣
 ↪netloc are present
    except ValueError:
        return False
```

[35]:
```
is_valid_url("http://localhost:8888/notebooks/Untitled1.ipynb")
```

[35]:
```
True
```

[38]:
```
#35. Extract the protocol (http or https) from a URL string.
def extract_protocol(url):
    protocol_end=url.find("://")
    if protocol_end != -1:
        protocol = url[:protocol_end]
        return protocol
    else :
        return none
```

[40]:
```
extract_protocol("http://www.example.com")
```

[40]:
```
'http'
```

[49]:
```
#36. Find the frequency of each character in a string.
str = input("enter string")
l=list(str)
freq = [l.count(ele) for ele in l]
d=dict(zip(l,freq))
print(d)
```

```
enter stringvij
{'v': 1, 'i': 1, 'j': 1}
```

[87]:
```
#37.Remove all punctuation from a string.
punctuations=''' !@#$%^&*(){}|:" <>?/.,;[]\ '''
my_str = input("enter string:")
no_punct=""
for char in my_str:
    if char not in punctuations:
        no_punct = no_punct+char
print(no_punct)
```

```
enter string:ggggh@!#$%^&
ggggh
```

[95]:
```
#38.Check if a string contains only digits.
string = (input("enter string"))
string.isdigit()
```

```
enter string1233
```

[95]:
```
True
```

[98]:
```
#39.Check if a string contains only alphabets.
string= (input("enter string"))
string.isalpha()
```

```
enter stringqwwee1234
```

[98]:
```
False
```

[101]:
```
#40.Convert a string to a list of characters.
string= (input("enter string"))
list(string)
```

```
enter stringvhgh
```

[101]:
```
['v', 'h', 'g', 'h']
```

[107]:
```
#41. Check if two strings are anagrams.
string1="lentis"
string2= "silent"
sorted(string1) == sorted(string2)
```

[107]: True

[128]:
```python
#42. Encode a string using a Caesar cipher.

def caesar_cipher(text, shift):
  """
  Encodes a string using a Caesar cipher.

  Args:
    text: The string to encode.
    shift: The number of positions to shift each letter.

  Returns:
    The encoded string.
  """

  # Initialize an empty string to store the encoded text.
  encoded_text = ""

  # Iterate through each character in the input text.
  for char in text:
    # Check if the character is alphabetic.
    if char.isalpha():
      # Calculate the encoded character's Unicode code point value by shifting it by the specified amount.
      # Ensure the value wraps around within the lowercase alphabet range (a-z).
      encoded_char = chr((ord(char) + shift - ord('a')) % 26 + ord('a'))
    else:
      # Keep non-alphabetic characters unchanged.
      encoded_char = char

    # Add the encoded character to the encoded text.
    encoded_text += encoded_char

  # Return the resulting encoded text.
  return encoded_text

# Define the input text and the shift value for encoding.
text = "Hello, world!"
shift = 3

# Call the caesar_cipher function to encode the text using the specified shift.
encoded_text = caesar_cipher(text, shift)

# Print the encoded text.
print("Encoded text:", encoded_text)
```

```
Encoded text: ehoor, zruog!
```

[129]:
```python
#43. Decode a Caesar cipher encoded string.

def caesar_cipher_decoder(text, shift):
    """
    Decodes a string using a Caesar cipher.

    Args:
        text: The string to decode.
        shift: The number of positions to shift each letter.

    Returns:
        The decoded string.
    """

    # Initialize an empty string to store the decoded text
    decoded_text = ""

    # Loop through each character in the input text
    for char in text:
        # Check if the character is an alphabetic character
        if char.isalpha():
            # Calculate the decoded character using the shift value
            decoded_char = chr((ord(char) - shift - ord('a')) % 26 + ord('a'))
        else:
            # If the character is not alphabetic, keep it unchanged
            decoded_char = char

        # Add the decoded character to the decoded text
        decoded_text += decoded_char

    # Return the fully decoded text
    return decoded_text

# Example input
text = "lopp, zruog!"
shift = 3

# Call the caesar_cipher_decoder function to decode the input text
decoded_text = caesar_cipher_decoder(text, shift)

# Print the decoded text
print("Decoded text:", decoded_text)
```

```
Decoded text: ilmm, world!
```

[2]:
```python
#44.Find the most frequent word in a string.
import re
from collections import Counter
```

```
def most_frequent_word(input_string):
    clean_string = re.sub(r'[^\w\s]', '', input_string).lower()
    words = clean_string.split()
    word_counter = Counter(words)
    most_common_word = word_counter.most_common(1)[0][0]
    return most_common_word
```

[4]:
```
most_frequent_word("Hello world, hello there! How's the world? hi hi hi hi")
```

[4]:
```
'hi'
```

[1]:
```
#45. Find all unique words in a string
str = input("enter a string").split()
l=list()
for ele in str:
    if ele not in l:
        l.append(ele)
print("unique words are",l)
```

```
enter a stringvv hh vv vin
unique words are ['vv', 'hh', 'vin']
```

[2]:
```
#46.Count the number of syllables in a string
import re

def count_syllables(word):
    # Count vowel groups using regular expression
    vowel_groups = re.findall(r'[aeiouy]+', word, re.IGNORECASE)

    return len(vowel_groups)

# Example words
word1 = "hello"
word2 = "banana"
word3 = "syllable"

# Count syllables
syllables1 = count_syllables(word1)
syllables2 = count_syllables(word2)
syllables3 = count_syllables(word3)

# Print the results
print(f"'{word1}' has {syllables1} syllables.")
print(f"'{word2}' has {syllables2} syllables.")
print(f"'{word3}' has {syllables3} syllables.")
```

```
'hello' has 2 syllables.
'banana' has 3 syllables.
```

```
'syllable' has 3 syllables.
```

[5]:
```python
#47.Check if a string contains any special characters.
import re
str = input("enter string")
regex = re.compile('[@_!#$%^&*()<>?/\}{~:]')

if(regex.search(str)== None):
    print("no special characters")
else:
    print("special characters are present in string")
```

```
enter stringdfghj
no special characters
```

[5]:
```python
#48.Remove the nth word from a string.
def remove_nth_word(string, n):
    words = string.split()

    if n < 1 or n > len(words):
        raise ValueError(f"n must be between 1 and {len(words)}")

    return " ".join(words[:n-1] + words[n:])

# Example usage
original_string = "This is a string with five words."
n = 3

modified_string = remove_nth_word(original_string, n)

print(f"Original string: {original_string}")
print(f"Modified string: {modified_string}")
```

```
Original string: This is a string with five words.
Modified string: This is string with five words.
```

[3]:
```python
# 49.Insert a word at the nth position in a string.
def insert_word_at_nth_position(string, n, word):
    if n < 1:
        raise ValueError("n must be a positive integer")

    if n > len(string) + 1:
        raise ValueError("n exceeds string length")

    return string[:n-1] + word + string[n-1:]

# Example usage
original_string = "This is a string."
```

```
n = 4
word = "new "

modified_string = insert_word_at_nth_position(original_string, n, word)

print(f"Original string: {original_string}")
print(f"Modified string: {modified_string}")
```

```
Original string: This is a string.
Modified string: Thinew s is a string.
```

[13]:
```
# 50. Convert a CSV string to a list of lists.
def csv_string_to_list(csv_string):
    # Split the CSV string into lines
    lines = csv_string.split('\n')

    # Initialize an empty list to store the result
    result = []

    for line in lines:
        # Split each line into fields using comma as the delimiter
        fields = line.split(',')

        # Append the fields to the result list as a row
        result.append(fields)

    return result

# Example CSV string
csv_data = "Name, Age, Country\nJohn, 25, USA\nAlice, 30, Canada\nBob, 22, UK"

# Convert CSV string to a list of lists
list_of_lists = csv_string_to_list(csv_data)

# Print the result
for row in list_of_lists:
    print(row)
```

```
['Name', ' Age', ' Country']
['John', ' 25', ' USA']
['Alice', ' 30', ' Canada']
['Bob', ' 22', ' UK']
```

[6]:
```
# List Based Practice Problems
```

[8]:
```
#1.create a list with numbers from 1 to 100

numbers_list = list(range(1, 101))
```

[14]:
```
numbers_list
```

[14]:
```
[1,
 2,
 3,
 4,
 5,
 6,
 7,
 8,
 9,
 10,
 11,
 12,
 13,
 14,
 15,
 16,
 17,
 18,
 19,
 20,
 21,
 22,
 23,
 24,
 25,
 26,
 27,
 28,
 29,
 30,
 31,
 32,
 33,
 34,
 35,
 36,
 37,
 38,
 39,
 40,
 41,
 42,
 43,
 44,
 45,
```

46,
47,
48,
49,
50,
51,
52,
53,
54,
55,
56,
57,
58,
59,
60,
61,
62,
63,
64,
65,
66,
67,
68,
69,
70,
71,
72,
73,
74,
75,
76,
77,
78,
79,
80,
81,
82,
83,
84,
85,
86,
87,
88,
89,
90,
91,
92,

```
 93,
 94,
 95,
 96,
 97,
 98,
 99,
 100]
```

[21]:
```
#2.Find the length of a list without using the `len()` function
numbers = [1,2,3,4,5,6]
count = 0
for element in numbers:
    count += 1

print("length of the list is:",count)
```

```
length of the list is: 6
```

[22]:
```
#3. Append an element to the end of a list.
numbers = [1,2,3,4,5,6]
numbers.append(7)
```

[23]:
```
numbers
```

[23]:
```
[1, 2, 3, 4, 5, 6, 7]
```

[26]:
```
#4.Insert an element at a specific index in a list
l=[1,2,3,4]
l.insert(2,5)
print(l)
```

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

[28]:
```
# 5. Remove an element from a list by its value.
l=[1,2,3,4,5,6]
l.remove(3)
print(l)
```

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

[29]:
```
#6. Remove an element from a list by its index.
l=[6,7,8,9,0]
l.pop(2)
print(l)
```

```
[6, 7, 9, 0]
```

[32]:
```
# 7. Check if an element exists in a list.
l=[6,7,8,9,0]
i=8
if i in l:
    print(True)
else:
    print(False)
```

```
True
```

[6]:
```
# 8. Find the index of the first occurrence of an element in a list.
l=[6,7,8,9,0,8]
i=6
index=l.index(9)
print(index)
```

```
3
```

[8]:
```
# 9. Count the occurrences of an element in a list.
l=[6,7,8,9,0,8,8,8]
i=8
countno=l.count(8)
print(countno)
```

```
4
```

[36]:
```
# 10. Reverse the order of elements in a list.
l=[6,7,8,9,0,8]
l.reverse()
print(l)
```

```
[8, 0, 9, 8, 7, 6]
```

[38]:
```
#11. Sort a list in ascending order.
l=[6,7,8,9,0,8]
sortedl=sorted(l)
print(sortedl)
```

```
[0, 6, 7, 8, 8, 9]
```

[39]:
```
#12. Sort a list in descending order.
def sort_list_desc(list1):
    return sorted(list1,reverse=True)
```

[40]:
```
sort_list_desc([6,7,8,9,0,8])
```

[40]:
```
[9, 8, 8, 7, 6, 0]
```

[41]:
```
#13.Create a list of even numbers from 1 to 20.
l=[i for i in range(1,21) if i % 2==0 ]
print(l)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

[43]:
```
#14.Create a list of odd numbers from 1 to 20.
l=[i for i in range(1,21) if i %2 !=0 ]
print(l)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

[46]:
```
# 15. Find the sum of all elements in a list.
l=[6,7,8,9,0,8,10]
total = sum(l)
print(total)
```

```
48
```

[47]:
```
# 16. Find the maximum value in a list.
l=[6,7,8,9,0,8,10]
maximum=max(l)
print(maximum)
```

```
10
```

[48]:
```
#17. Find the minimum value in a list.
l=[6,7,8,9,0,8,10]
minimum=min(l)
print(minimum)
```

```
0
```

[52]:
```
#18. Create a list of squares of numbers from 1 to 10.
squares = [x*x for x in range (1,11)]
print(squares)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

[12]:
```
#19.Create a list of random numbers.
import random
random_list= [random.randint(1,100) for i in range(10)]
print(random_list)
```

```
[45, 22, 7, 59, 82, 52, 80, 12, 63, 7]
```

[11]:
```
#20. Remove duplicates from a list
l=[6,7,8,9,0,8,10,6,7,8,6,8]
l=set(l)
print(l)
```

```
{0, 6, 7, 8, 9, 10}
```

[70]:
```
#21.Find the common elements between two lists.
list1= [1,2,3,4,5,6]
list2= [1,2,3]
commom_elements = set(list1)&set(list2)
print("common elements",commom_elements)
```

```
common elements {1, 2, 3}
```

[72]:
```
# 22. Find the difference between two lists
list1 = [1,2,3,4,5]
list2 = [5,3,8]

difference = [x for x in list1 if x not in list2]
print(difference)
```

```
[1, 2, 4]
```

[16]:
```
# 23. Merge two lists.
list1 =[1,2,3,4,5]
list2 = [6,7,8,9]
merge_list = list1 +list2
print(merge_list)
```

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

[75]:
```
# 24. Multiply all elements in a list by 2.
list1=[1,2,3,4,5,6]
multiplied_list = [ i*2 for i in list1 ]
print(multiplied_list)
```

```
[2, 4, 6, 8, 10, 12]
```

[77]:
```
# 25. Filter out all even numbers from a list.
list1 = [i for i in range(1,11) if i%2 != 0]
print(list1)
```

```
[1, 3, 5, 7, 9]
```

[3]:
```
# 26. Convert a list of strings to a list of integers.
string1 = ["1","2","3","4"]
int_str= [ int(x)for x in string1 ]
print(int_str)
```

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

[3]:
```
# 27. Convert a list of integers to a list of strings.
integers = [1,2,3,4]
stri = [str(x) for x in integers]
print(stri)
```

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

[4]:
```python
# 28. Flatten a nested list.
nested_list = [[1, 2, 3], [4, 5], [6]]
flat_list = sum(nested_list, [])
print(flat_list)
```

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

[13]:
```python
#29.Create a list of the first 10 Fibonacci numbers.
def fibonacci(n):
    if n <= 1:
        return  n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
fibonacci_numbers = [fibonacci(i) for i in range(10)]
print(fibonacci_numbers)
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

[14]:
```python
#30. Check if a list is sorted.
def is_sorted(data):
    for i in range(1, len(data)):
        if data[i-1] > data[i]:
            return False
    return True
```

[15]:
```python
is_sorted([1,2,3,4])
```

[15]:
```
True
```

[17]:
```python
#31.Rotate a list to the left by `n` positions.
def left_rotate(lst, n):
    n = n % len(lst)
    return lst[n:] + lst[:n]
# Example usage
my_list = [1, 2, 3, 4, 5]
n = 2  # Rotate two positions to the left

rotated_list = left_rotate(my_list, n)
print(rotated_list)
```

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

[20]:
```python
#32.Rotate a list to the right by `n` positions.
def right_rotate(lst, n):
    n = n % len(lst)
    return lst[-n:] + lst[:-n]
# Example usage
my_list = [1, 2, 3, 4, 5]
n = 2  # Rotate two positions to the right
```

```
rotated_list = right_rotate(my_list, n)
print(rotated_list)
```

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

[21]:
```
#33.Create a list of prime numbers up to 50
prime_numbers = [n for n in range(2, 51) if all(n % i != 0 for i in range(2,␣
 ↪int(n**0.5) + 1))]

print(prime_numbers)
```

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

[34]:
```
#34.Split a list into chunks of size `n`.
def chunks(lst,n):
    for i in range(0,len(lst),n):
        yield lst[i:i+n]
mylist=[1,2,3,4,5,6,7,8]
chunks1=list(chunks(mylist,3))
print(chunks1)
```

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

[35]:
```
#35.Find the second largest number in a list
def second_largest(data):
    if len(data)<2 :
        return none
    sorted_data = sorted(data)
    return sorted_data[-2]
```

[39]:
```
second_largest([2,5,7,1,6])
```

[39]:
```
6
```

[46]:
```
# 36. Replace every element in a list with its square.
l = [1,2,3,4,5]
v = [x*x for x in l ]
print(v)
```

```
[1, 4, 9, 16, 25]
```

[47]:
```
# 37. Convert a list to a dictionary where list elements become keys and their␣
 ↪indices become values.
l = [1,2,3,4,5]
v = dict(enumerate(l))
print(v)
```

```
{0: 1, 1: 2, 2: 3, 3: 4, 4: 5}
```

[4]:
```python
# 38. Shuffle the elements of a list randomly.
import random
l = [1,2,3,4,5]
random.shuffle(l)
print(l)
```

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

[7]:
```python
# 39. Create a list of the first 10 factorial numbers
factorials=[]
for n in range(1,11):
    factorial=1
    for i in range(1,n+1):
        factorial *= i

    factorials.append(factorial)
print(factorials)
```

```
[1, 2, 6, 24, 120, 720, 5040, 40320, 362880, 3628800]
```

[8]:
```python
#40.Check if two lists have at least one element in common.
list1=[1,2,3,4]
list2=[5,6,7,1]
v = any(item in list2 for item in list1)
print(v)
```

```
True
```

[12]:
```python
#41. Remove all elements from a list.
l1=[1,2,3,4,5]
l1.clear()
print(l1)
```

```
[]
```

[16]:
```python
#42Replace negative numbers in a list with 0.
l1=[1,-2,-3,4]
l1=[0 if num < 0 else num for num in l1]
print(l1)
```

```
[1, 0, 0, 4]
```

[17]:
```python
# 43. Convert a string into a list of words.
v = "deepu is good"
d=v.split()
print(d)
```

```
['deepu', 'is', 'good']
```

[18]:
```
# 44. Convert a list of words into a string.
words =['deepu', 'is', 'good']
sentence =" ".join(words)
print(sentence)
```

```
deepu is good
```

[22]:
```
#45. Create a list of the first `n` powers of 2
l=[1,2,3,4]
v=[2**x for x in l]
print(v)
```

```
[2, 4, 8, 16]
```

[23]:
```
#46. Find the longest string in a list of strings.
strings = ["This", "is", "a", "list", "of", "strings", "with", "different",␣
 ↪"lengths"]
longs=max(strings,key=len)
print(longs)
```

```
different
```

[24]:
```
#47. Find the shortest string in a list of strings.
strings = ["This", "is", "a", "list", "of", "strings", "with", "different",␣
 ↪"lengths"]
shorts=min(strings,key=len)
print(shorts)
```

```
a
```

[93]:
```
#48. Create a list of the first `n` triangular numbers.
l1= [i*(i+1)//2 for i in range(5)]
print(l1)
```

```
[0, 1, 3, 6, 10]
```

[34]:
```
# 49. Check if a list contains another list as a subsequence.
def is_subsequence(list1, list2):

    """Checks if list2 is a subsequence of list1 using a loop."""
    i = 0
    for item in list1:
        if item == list2[i]:
            i += 1
            if i == len(list2):
                return True
    return False
```

[39]:
```
is_subsequence([1,2,3,4],[])
```

[39]: True

[43]:
```python
# 50. Swap two elements in a list by their indices.
def swap_elements_unpacking(list1, index1, index2):
    """Swaps two elements in a list using tuple unpacking."""
    list1[index1], list1[index2] = list1[index2], list1[index1]

# Example usage:
my_list = [10, 20, 30, 40, 50]
index1 = 1
index2 = 3

swap_elements_unpacking(my_list, index1, index2)
print(my_list)
```

```
[10, 40, 30, 20, 50]
```

[1]:
```python
# Tuple based practice problems
```

[2]:
```python
#1. Create a tuple with integers from 1 to 5.
v = (1,2,3,4,5)
print(v)
```

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

[3]:
```python
# 2. Access the third element of a tuple.
v = (1,2,3,4,5)
v[2]
```

[3]: 3

[5]:
```python
# 3. Find the length of a tuple without using the `len()` function.
v = (1,2,3,4,5,0)
count=0
for ele in v:
    count +=1

print(count)
```

```
6
```

[8]:
```python
# 4. Count the occurrences of an element in a tuple.
v = (1,2,3,4,5,0,1,1)
countno = v.count(1)
print(countno)
```

```
3
```

[11]:
```
# 5. Find the index of the first occurrence of an element in a tuple.
v = (1,2,3,4,5,0,1,1)
count_index=v.index(4)
print(count_index)
```

```
3
```

[18]:
```
# 6. Check if an element exists in a tuple
v = (1,2,3,4,5,0,1,1)
element=9
if element in v:
    print(True)
else:
    print(False)
```

```
False
```

[20]:
```
# 7. Convert a tuple to a list.
v = (1,2,3,4,5,0,1,1)
a=list(v)
print(a)
```

```
[1, 2, 3, 4, 5, 0, 1, 1]
```

[21]:
```
# 8. Convert a list to a tuple.
v= [1, 2, 3, 4, 5, 0, 1, 1]
a=tuple(v)
print(a)
```

```
(1, 2, 3, 4, 5, 0, 1, 1)
```

[23]:
```
# 9. Unpack the elements of a tuple into variables.
v= (1, 2, 3, 4, 5)
a,b,c,d,e=v
print(f"a:1")
print(f"b:2")
print(f"c:3")
print(f"d:4")
print(f"e:5")
```

```
a:1
b:2
c:3
d:4
e:5
```

[26]:
```
# 10. Create a tuple of even numbers from 1 to 10.
l=tuple(i for i in range(1,11) if i % 2==0 )
print(l)
```

```
(2, 4, 6, 8, 10)
```

[33]:
```
#11. Create a tuple of odd numbers from 1 to 10
v=tuple(i for i in range(1,10) if i%2!= 0)
print(v)
```

```
(1, 3, 5, 7, 9)
```

[34]:
```
# 12. Concatenate two tuples.
v=(1, 3, 5, 7, 9)
a=(2, 4, 6, 8, 10)
d=v+a
print(d)
```

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

[39]:
```
# 13. Repeat a tuple three times.
v=(1, 3, 5, 7, 9)
a=v*3
print(a)
```

```
(1, 3, 5, 7, 9, 1, 3, 5, 7, 9, 1, 3, 5, 7, 9)
```

[43]:
```
# 14. Check if a tuple is empty
v=()
if len(v) == 0:
    print("empty")
else:
    print("notempty")
```

```
empty
```

[50]:
```
# 15. Create a nested tuple.
v=((1, 3, 5),(7, 9, 1, 3),( 5, 7, 9, 1,3, 5, 7, 9))
print(v)
```

```
((1, 3, 5), (7, 9, 1, 3), (5, 7, 9, 1, 3, 5, 7, 9))
```

[60]:
```
# 16. Access the first element of a nested tuple.
v=((1, 3, 5),(7, 9, 1, 3),( 5, 7, 9, 1,3, 5, 7, 9))
a=v[0][0]
print(a)
```

```
1
```

[64]:
```
# 17. Create a tuple with a single element
v=(1,)
print(v)
print(type(v))
```

```
(1,)
<class 'tuple'>
```

[67]:
```
# 18. Compare two tuples.
v=(1, 2,3)
a=(1,2,3,4)
v==a
```

[67]:
```
False
```

[76]:
```
# 19. Delete a tuple
v=(1, 2,3)
del v
print(v)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[76], line 4
      2 v=(1, 2,3)
      3 del v
----> 4 print(v)

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

[81]:
```
# 20. Slice a tuple
v=(1,2,3,4,5,6)
v[0:5:2]
```

[81]:
```
(1, 3, 5)
```

[82]:
```
# 21. Find the maximum value in a tuple.

v=(1,2,3,4,5,6)
max(v)
```

[82]:
```
6
```

[83]:
```
# 22. Find the minimum value in a tuple.
v=(1,2,3,4,5,6)
min(v)
```

[83]:
```
1
```

[88]:
```
# 23. Convert a string to a tuple of characters.
v='hello'
tuple(v)
```

[88]: ('h', 'e', 'l', 'l', 'o')

[90]:
```
# 24. Convert a tuple of characters to a string
v=('h', 'e', 'l', 'l', 'o')
a=" ".join(v)
print(a)
```

```
h e l l o
```

[91]:
```
# 25. Create a tuple from multiple data types.
a=("hello",3.14,True,11)
print(a)
```

```
('hello', 3.14, True, 11)
```

[10]:
```
# 26. Check if two tuples are identical.
a=(1,2,3)
b=(2,1,3)
a == b
```

[10]: False

[12]:
```
# 27. Sort the elements of a tuple.
v=(6,9,8,1,5,5,2)
a= sorted(v)
print(a)
```

```
[1, 2, 5, 5, 6, 8, 9]
```

[15]:
```
# 28. Convert a tuple of integers to a tuple of strings.
v=(6,9,8,1,5,5,2)
a=tuple([str(x) for x in v ])
print(a)
```

```
('6', '9', '8', '1', '5', '5', '2')
```

[19]:
```
# 29. Convert a tuple of strings to a tuple of integers.
v=('1','2','3')
a=tuple(map(int,v))
print(a)
```

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

[23]:
```
# 30. Merge two tuples.
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)

merged_tuple = tuple1 + tuple2

print(f"Original tuple 1: {tuple1}")
```

```
print(f"Original tuple 2: {tuple2}")
print(f"Merged tuple: {merged_tuple}")
```

```
Original tuple 1: (1, 2, 3)
Original tuple 2: (4, 5, 6)
Merged tuple: (1, 2, 3, 4, 5, 6)
```

[37]:
```
# 31. Flatten a nested tuple.
a=((1,2,3),(4,5,6))
b=a[0]+a[1]
print(b)
```

```
(1, 2, 3, 4, 5, 6)
```

[38]:
```
# 32. Create a tuple of the first 5 prime numbers
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

primes = ()  # Start with an empty tuple
count = 0
num = 2
while count < 5:
    if is_prime(num):
        primes += (num,)  # Add the prime number to the tuple
        count += 1
    num += 1

print(primes)
```

```
(2, 3, 5, 7, 11)
```

[47]:
```
# 33. Check if a tuple is a palindrome.
def is_palindrome(tupl):
    v=" ".join(str(x) for x in tupl)
    return v == v[::-1]

tupl1=("a", "b", "b", "a")
tupl2=("a", "b", "c", "d")
print(f"tuple {tupl1} is a palindrome:{is_palindrome(tupl1)}")
print(f"tuple {tupl2} is a palindrome:{is_palindrome(tupl2)}")
```

```
tuple ('a', 'b', 'b', 'a') is a palindrome:True
tuple ('a', 'b', 'c', 'd') is a palindrome:False
```

[52]:
```
# 34. Create a tuple of squares of numbers from 1 to 5.

v=tuple(x**2 for x in range(1,6))
print(v)
```

```
(1, 4, 9, 16, 25)
```

[53]:
```
#35.Filter out all even numbers from a tuple.
a=(1,2,3,4,5,6,7,8,9,10)
v=tuple(x for x in a if x%2 !=0)
print(v)
```

```
(1, 3, 5, 7, 9)
```

[54]:
```
# 36. Multiply all elements in a tuple by 2
a=(1,2,3,4,5,6,7,8,9,10)
v=tuple(x*2 for x in a)
print(v)
```

```
(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)
```

[58]:
```
# 37. Create a tuple of random numbers.
import random

v=tuple(random.randint(0,10) for i in range(5))
print(v)
```

```
(1, 10, 7, 3, 2)
```

[65]:
```
# 38. Check if a tuple is sorted
def is_sorted(tupl):
    for i in range(len(tupl)-1):
        if tupl[i]>tupl[i+1]:
            return False
    return True
```

[66]:
```
is_sorted((1, 10, 7, 3, 2))
```

[66]:
```
False
```

[67]:
```
# 39. Rotate a tuple to the left by `n` positions.
def rotate_left(tupl,n):
    a= tupl[n:]+tupl[:n]
    return a
```

[68]:
```
rotate_left((1, 10, 7, 3, 2),2)
```

[68]:
```
(7, 3, 2, 1, 10)
```

[71]:
```
# 40. Rotate a tuple to the right by `n` positions.
def roatate_right(tupl,n):
    a=tupl[-n:]+tupl[:-n]
    return a
```

[72]:
```
roatate_right((1, 10, 7, 3, 2),2)
```

[72]:
```
(3, 2, 1, 10, 7)
```

[79]:
```
# 41. Create a tuple of the first 5 Fibonacci numbers
def fibonacci(n):
  """Generates the nth Fibonacci number recursively."""
  if n <= 1:
    return n
  else:
    return fibonacci(n - 1) + fibonacci(n - 2)

# Print the first 5 Fibonacci numbers

for i in range(5):

    print(fibonacci(i))
```

```
0
1
1
2
3
```

[80]:
```
# 42. Create a tuple from user input.
numbers = input("enter number seperated by spaces")
a = tuple(numbers.split())
print(a)
```

```
enter number seperated by spaces(1,2,3)
('(1,2,3)',)
```

[85]:
```
#43.Swap two elements in a tuple.
v=(10,20,30,40)
l=list(v)
l[0],l[1]=l[2],l[3]
v=tuple(l)
print(v)
```

```
(30, 40, 30, 40)
```

[86]:
```
# 44. Reverse the elements of a tuple
v=(10,20,30,40)
l=list(v)
```

```
l=l[::-1]
v=tuple(l)
print(v)
```

```
(40, 30, 20, 10)
```

[88]:
```
# 45. Create a tuple of the first `n` powers of 2.
v=(10,20,30,40)
l=list(v)
a=[2**x for x in l]
b=tuple(a)
print(b)
```

```
(1024, 1048576, 1073741824, 1099511627776)
```

[89]:
```
# 46. Find the longest string in a tuple of strings.
strings = ("This", "is", "a", "list", "of", "strings", "with", "different",␣
 ↪"lengths")
longs=max(strings,key=len)
print(longs)
```

```
different
```

[90]:
```
# 47. Find the shortest string in a tuple of strings.
strings=("hi","hello","a")
v=min(strings,key=len)
print(v)
```

```
a
```

[95]:
```
# 48. Create a tuple of the first `n` triangular numbers.
l1= tuple(i*(i+1)//2 for i in range(5))
print(l1)
```

```
(0, 1, 3, 6, 10)
```

[96]:
```
# 49. Check if a tuple contains another tuple as a subsequence.
def is_subsequence(main_tupl,sub_tupl):
    sub_iter=iter(sub_tupl)
    return all(elem in main_tupl for elem in sub_tupl)
```

[100]:
```
is_subsequence((1,2,3,4,5),(6,))
```

[100]:
```
False
```

[103]:
```
# 50. Create a tuple of alternating 1s and 0s of length `n`
def alternating_tuple(n):
    return tuple(1 if i% 2 else 0 for i in range(n))
```

[104]:
```
alternating_tuple(6)
```

[104]: (0, 1, 0, 1, 0, 1)

[1]:
```
# set based practice problems
```

[2]:
```
#1. Create a set with integers from 1 to 5
a={1,2,3,4,5}
print(a)
```

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

[4]:
```
# 2. Add an element to a set
a={1,2,3,4,5}
a.add(6)
print(a)
```

```
{1, 2, 3, 4, 5, 6}
```

[5]:
```
# 3. Remove an element from a set.
a={1,2,3,4,5}
a.remove(5)
print(a)
```

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

[12]:
```
# 4. Check if an element exists in a set
a={1,2,3,4,5}

if 1 in a:
    print(True)
else:
    print(False)
```

```
True
```

[21]:
```
# 5. Find the length of a set without using the `len()` function.
a={1,2,3,4,5}
count=0
for i in a:
    count += 1

print(count)
```

```
5
```

[22]:
```
# 6. Clear all elements from a set.
a={1,2,3,4,5}
a.clear()
print(a)
```

```
set()
```

[23]:
```
# 7. Create a set of even numbers from 1 to 10
v=set(x for x in range(10) if x%2 ==0)
print(v)
```

```
{0, 2, 4, 6, 8}
```

[26]:
```
# 8. Create a set of odd numbers from 1 to 10.
v=set(x for x in range(10) if x%2!=0)
print(v)
```

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

[29]:
```
# 9. Find the union of two sets.
a={1,2,3,5}
b={4,6,7,8}
a.union(b)
```

[29]:
```
{1, 2, 3, 4, 5, 6, 7, 8}
```

[31]:
```
# 10. Find the intersection of two sets.
a={1,2,3,5,6,7}
b={4,6,7,8}
b.intersection(a)
```

[31]:
```
{6, 7}
```

[32]:
```
# 11. Find the difference between two sets.
a={1,2,3,4,5}
b={4,6,7,8}
a-b
```

[32]:
```
{1, 2, 3, 5}
```

[37]:
```
# 12. Check if a set is a subset of another set.
a={1,2,3,4,5}
b={4,5}
b.issubset(a)
```

[37]:
```
True
```

[39]:
```
# 13. Check if a set is a superset of another set.
a={1,4,5}
b={1,7,8,4,9,6,5}
b.issuperset(a)
```

[39]:
```
True
```

[41]:
```
# 14. Create a set from a list.
l=[1,2,3,4]
```

```
set(l)
```

[41]: {1, 2, 3, 4}

[42]:
```
# 15. Convert a set to a list
s={1, 2, 3, 4}
list(s)
```

[42]: [1, 2, 3, 4]

[52]:
```
# 16.Remove a random element from a set.
import random
s = {1,2,3,4,5,6}
random_no=random.choice(list(s))
s.remove(random_no)
print(random_no)
```

```
1
```

[56]:
```
# 17. Pop an element from a set
s = {1,2,3,4,5,6}
a=s.pop()
print(a)
print(s)
```

```
1
{2, 3, 4, 5, 6}
```

[70]:
```
# 18. Check if two sets have no elements in common.
a={1,2,3,4,5}
b={1,2,3}
print(a.intersection(b))
if not a.intersection(b):
    print("no ele are common")
else:
    print("ele are common")
```

```
{1, 2, 3}
ele are common
```

[77]:
```
# 19. Find the symmetric difference between two sets.
a={1,2,3,4,5}
b={4,5,6}
symmetric_difference=a.symmetric_difference(b)
print(symmetric_difference)
```

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

[79]:
```
# 20. Update a set with elements from another set.
a={1,2,3,4,5}
b={4,5,6}
a.update(b)
print(a)
```

```
{1, 2, 3, 4, 5, 6}
```

[86]:
```
# 21. Create a set of the first 5 prime numbers
prime_numbers = set()

# Check numbers from 2 to 5 (inclusive)
for num in range(2, 6):
  # Check if the number is prime
  is_prime = True

  for i in range(2, num):
    if num % i == 0:
      is_prime = False
      break
  # Add the prime number to the set if it is
  if is_prime:
    prime_numbers.add(num)

# Print the set of prime numbers
print(f"The first 5 prime numbers are: {prime_numbers}")
```

```
The first 5 prime numbers are: {2, 3, 5}
```

[87]:
```
# 22. Check if two sets are identical.
set1={1,2,3}
set2={1,2,3}
if set1==set2 :
    print("sets  are identical")
else:
    print("sets are not identical")
```

```
sets  are identical
```

[88]:
```
# 23. Create a frozen set
s={1,2,3,4,5,1,2,3}
frozen_no = frozenset(s)
print(frozen_no)
```

```
frozenset({1, 2, 3, 4, 5})
```

[91]:
```
# 24. Check if a set is disjoint with another set.
set1={1,2,3}
set2={4,5,6}
```

```
set1.isdisjoint(set2)
```

[91]: True

[93]:
```
# 25. Create a set of squares of numbers from 1 to 5.
a={1,2,3,4,5}
v=set(x*x for x in a)
print(v)
```

```
{1, 4, 9, 16, 25}
```

[94]:
```
# 26. Filter out all even numbers from a set.
v=set(x for x in range(10) if x%2!=0)
print(v)
```

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

[95]:
```
# 27. Multiply all elements in a set by 2.
a={1,2,3,4,5}
v=set(x*2 for x in a)
print(v)
```

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

[100]:
```
# 28. Create a set of random numbers
import random
random_no=set(random.sample(range(1,11),5))
print(random_no)
```

```
{3, 4, 7, 8, 9}
```

[102]:
```
# 29. Check if a set is empty
a={}
if len(a) ==0:
    print("empty")
else:
    print("notempty")
```

```
empty
```

[114]:
```
# 30. Create a nested set (hint: use frozenset).
nested_set = {frozenset({1, 2, 3}), frozenset({'a', 'b', 'c'}), frozenset({4.5,␣
↪6.7, 8.9})}

print(nested_set)
```

```
{frozenset({1, 2, 3}), frozenset({8.9, 4.5, 6.7}), frozenset({'c', 'a', 'b'})}
```

[116]:
```
# 31. Remove an element from a set using the discard method.
```

```
a = {1,2,3,4}
a.discard(1)
print(a)
```

```
{2, 3, 4}
```

[117]:
```
# 32. Compare two sets.
a = {1,2,3,4}
b={5,6,7,8}
a==b
```

[117]:
```
False
```

[122]:
```
# 33. Create a set from a string
string=('hello')
a=set(string)
print(a)
```

```
{'o', 'h', 'e', 'l'}
```

[127]:
```
# 34. Convert a set of strings to a set of integers.
strings={'1','2','3'}
integers=set(int(x) for x in strings)

print(integers)
```

```
{1, 2, 3}
```

[128]:
```
# 35. Convert a set of integers to a set of strings.
integers={1,2,3}
strings=set(str(x) for x in integers)
print(strings)
```

```
{'3', '2', '1'}
```

[129]:
```
# 36. Create a set from a tuple.
tupl=(1,2,3)
set1=set(tupl)
print(set1)
```

```
{1, 2, 3}
```

[130]:
```
# 37. Convert a set to a tuple.
set1={1,2,3}
tupl=tuple(set1)
print(tupl)
```

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

[131]:
```
# 38. Find the maximum value in a set.
set1={1,2,3}
a=max(set1)
print(a)
```

```
3
```

[132]:
```
# 39. Find the minimum value in a set.
set1={1,2,3}
a=min(set1)
print(a)
```

```
1
```

[136]:
```
# 40. Create a set from user input.
user_input=input("enter elements(comma seperated)")
elem=user_input.split(',')
a=set(elem)
print(a)
```

```
enter elements(comma seperated)1,2,3
{'3', '2', '1'}
```

[144]:
```
# 41. Check if the intersection of two sets is empty.
set1 = {1, 2, 3, 4}
set2 = {3, 4, 5, 6}

# Check if the intersection is empty
if set1.isdisjoint(set2):
    print("The intersection is empty.")
else:
    print("The intersection is not empty.")
```

```
The intersection is not empty.
```

[145]:
```
#42.Create a set of the first 5 Fibonacci numbers.
def generate_fibonacci(n):
    fibonacci_set = set()
    a, b = 0, 1

    for _ in range(n):
        fibonacci_set.add(a)
        a, b = b, a + b

    return fibonacci_set

# Create a set of the first 5 Fibonacci numbers
first_5_fibonacci = generate_fibonacci(5)
```

```
print("Set of the first 5 Fibonacci numbers:", first_5_fibonacci)
```

```
Set of the first 5 Fibonacci numbers: {0, 1, 2, 3}
```

[147]:
```
# 43. Remove duplicates from a list using sets.
l=[1,2,3,1,2,3,4,5,6,4,5,6]
v=set(l)
l1=list(v)
print(l1)
```

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

[149]:
```
# 44. Check if two sets have the same elements, regardless of their count.
a={1,2,3,4,5}
b={5,4,3,2,1}
if a==b :
    print("two sets have the same elements, regardless of their count")
else:
     print("two sets do not have the same elements, regardless of their count")
```

```
two sets have the same elements, regardless of their count
```

[150]:
```
# 45. Create a set of the first `n` powers of 2.
v=set(2**i for i in range(5))
print(v)
```

```
{1, 2, 4, 8, 16}
```

[151]:
```
# 46.Find the common elements between a set and a list.
a={1,2,3,4,5}
b=[1,2,3]
common_ele=set(a)& set(b)
print(common_ele)
```

```
{1, 2, 3}
```

[152]:
```
# 47. Create a set of the first `n` triangular numbers.
def traingular(n):
    return {i*i+1//2 for i in range(1,n+1)}
first_n_traingularnos=traingular(5)

print(first_n_traingularnos)
```

```
{1, 4, 9, 16, 25}
```

[154]:
```
# 48. Check if a set contains another set as a subset.
a={1,2,3,4,5}
b={4,5}
b.issubset(a)
```

[154]:
```
True
```

[163]:
```
#49. Create a set of alternating 1s and 0s of length `n`.
def alternating(n):
    return set(i%2 for i in range(n))
```

[164]:
```
alternating(6)
```

[164]:
```
{0, 1}
```

[165]:
```
# 50. Merge multiple sets into one.
a={1,2}
b={3,4}
c={5,6}
a.union(b,c)
```

[165]:
```
{1, 2, 3, 4, 5, 6}
```

[ ]: