dsm-english-batch-classes-day-4-1

February 1, 2023

1 Learnings and Notes of PWSKILLS DATA SCIENCE MASTERS CLASS DAY-4

1.1 Author:- ARPAN CHOUDHURY

[1]:
```
# String Manipulation Remaining section and functions.
# capitalizr\e()
name = "adeltech"
name.capitalize()
```

[1]:
```
'Adeltech'
```

[4]:
```
# Reversing a string
name[::-1] #using slicing
```

[4]:
```
<reversed at 0x1c83e7e9310>
```

[5]:
```
reversed(name)  # Returns a str object stored at specific memory location
```

[5]:
```
<reversed at 0x1c83faee730>
```

[14]:
```
# join() function
"".join(reversed(name)) # reversed(name) acts as an iterator
```

[14]:
```
'hcetleda'
```

[15]:
```
" ".join(reversed(name))
```

[15]:
```
'h c e t l e d a'
```

[8]:
```
"Abc".join("cde")
```

[8]:
```
'cAbcdAbce'
```

[9]:
```
" ".join("iNeuron")
```

[9]:
```
'i N e u r o n'
```

[12]:
```
list(reversed(name))
```

[12]:
```
['h', 'c', 'e', 't', 'l', 'e', 'd', 'a']
```

[16]:
```
## Removing characters from the end of the string
string_a = " PWSkills "
string_a.replace(" ",'')
```

[16]:
```
'PWSkills'
```

[17]:
```
string_a.strip()
```

[17]:
```
'PWSkills'
```

[18]:
```
string_a.lstrip()
```

[18]:
```
'PWSkills '
```

[20]:
```
string_a.rstrip()
```

[20]:
```
' PWSkills'
```

[19]:
```
string_b = " Happy Learning to All"
string_b.replace("to","from")
```

[19]:
```
' Happy Learning from All'
```

[21]:
```
# A new local var copy of the string is generated and all occurences of the␣
 ↪substr1 will be replaced by substr2
# where substr1 is substring to be replaced and substr2 is substring that␣
 ↪replaces substr1
print(id(string_b))
print(id(string_b.replace("to","from")))
```

```
1959581703424
1959581725840
```

[22]:
```
'Hello World'
```

[22]:
```
'Hello World'
```

[23]:
```
'Hello \tWorld'.expandtabs()
```

[23]:
```
'Hello   World'
```

[24]:
```
'Hello \nWorld'
```

[24]:
```
'Hello \nWorld'
```

[ ]:
```
'Hello \nWorld'
```

[ ]:
```
# for \n .format() can be used
```

[26]:
```
s1 = "Welcome to PWSkills "
```

[27]:
```
s1.islower()
```

[27]:
```
False
```

[28]:
```
s1.isupper()
```

[28]:
```
False
```

[31]:
```
"  ".isspace() # checks if all characters in the string are spaces
```

[31]:
```
True
```

[32]:
```
s2 = "iNeuron"
```

[33]:
```
s2.endswith('n')
```

[33]:
```
True
```

[34]:
```
s2.startswith('i')
```

[34]:
```
True
```

[35]:
```
s3 = "abcd123"
s3.isalnum()
```

[35]:
```
True
```

[38]:
```
c = 0
for i in s3:
    c+=1
print(c)
```

```
7
```

[39]:
```
len(s3)
```

[39]:
```
7
```

[40]:
```
vowels = "AaEeIiOoUu"
for i in s3:
    if i in vowels:
        print('{} is a vowel'.format(i))
    else:
        print("{} is not a vowel".format(i))
```

```
a is a vowel
b is not a vowel
c is not a vowel
d is not a vowel
1 is not a vowel
2 is not a vowel
3 is not a vowel
```

[42]:
```
for i in s3:
    print(i)
```

```
a
b
c
d
1
2
3
```

[43]:
```
for i in range(len(s3)):
    print(i,'->',s3[i])
```

```
0 -> a
1 -> b
2 -> c
3 -> d
4 -> 1
5 -> 2
6 -> 3
```

[44]:
```
for i in range(len(s3)-1,-1,-1):
    print(s3[i])
```

```
3
2
1
d
c
b
a
```

[109]:
```
z = [[1,2,3],[4,5,6],[7,8,9]]
[i[0] for i in z]
```

[109]:
```
[[1, 2, 3]]
```

2 LISTS

[45]:
```
["Knn","SVM","DSA",32]
```

[45]:
```
['Knn', 'SVM', 'DSA', 32]
```

[46]:
```
list([1,2,3,4,5,6])
```

[46]:
```
[1, 2, 3, 4, 5, 6]
```

[47]:
```
list(s3)
```

[47]:
```
['a', 'b', 'c', 'd', '1', '2', '3']
```

[52]:
```
str1 = "PW SKILLS DATA SCIENCE MASTERS"
lst1 = str1.split(" ")
```

[53]:
```
lst1[1:]
```

[53]:
```
['SKILLS', 'DATA', 'SCIENCE', 'MASTERS']
```

[54]:
```
lst1 = [1,3,4,5,6,7,8]
```

3 LIST FUNCTIONS

[55]:
```
lst1.reverse()
```

[56]:
```
lst1
```

[56]:
```
[8, 7, 6, 5, 4, 3, 1]
```

[57]:
```
lst1.sort()
```

[58]:
```
lst1
```

[58]:
```
[1, 3, 4, 5, 6, 7, 8]
```

[59]:
```
matrix = [[1,2,3],[4,5,6],[7,8,9]]
```

[62]:
```
matrix[0:][0]
```

[62]:
```
[1, 2, 3]
```

[173]:
```
lr = [[1,3,4],[5,6,7],[6,4,5]]
[i[0] for i in lr]
```

[173]:
```
[1, 5, 6]
```

[71]:
```
[i if i%2==0 else "ODD" for i in range(20)]
```

[71]:
```
[0,
 'ODD',
 2,
 'ODD',
 4,
 'ODD',
 6,
 'ODD',
 8,
 'ODD',
 10,
 'ODD',
 12,
 'ODD',
 14,
 'ODD',
 16,
 'ODD',
 18,
 'ODD']
```

[72]:
```
[i for i in range(20) if i%2==0]
```

[72]:
```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]
```

[128]:
```
l = [i for i in range(21)]
```

[77]:
```
odd_sum = sum([i for i in l if i%2==1])
odd_sum
```

[77]:
```
100
```

[ ]:
```
sq = [i*i for i in range(n)]
```

[129]:
```
# Create a list of prime numbers from A List using List Comprehension

# Create a list of
```

```
l1 = [i for i in l if sum([1 if i%j==0 else 0 for j in range(1,i+1)])==2 ]
```

[171]:
```
print(l)
print(l1)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
[2, 3, 5, 7, 11, 13, 17, 19]
```

[131]:
```
l2 = []
for i in l:
    c=0
    for j in range(1,i+1):
        if(i%j==0):
            c+=1
    if(c==2):
        l2.append(i)
```

[132]:
```
l2
```

[132]:
```
[2, 3, 5, 7, 11, 13, 17, 19]
```

[96]:
```
l1.remove(2)
```

[130]:
```
l1
```

[130]:
```
[2, 3, 5, 7, 11, 13, 17, 19]
```

[86]:
```
sum([1 if 5%j==0 else 0 for j in range(1,5+1)])
```

[86]:
```
2
```

[ ]:
```

```

[170]:
```
g=[(i,j) for i in range(len(l)-1) for j in range(i+1,len(l))] #Creation of␣
 ↪every possible combinations of 2 elements from list
# excluding identity combinations like (0,0), (1,1), .... (n,n) form if want to␣
 ↪include just use for j in range(i,len(l)):
print("Pairs possible are:\n",g)
```

```
Pairs possible are:
 [(0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (0, 7), (0, 8), (0, 9), (0,
10), (0, 11), (0, 12), (0, 13), (0, 14), (0, 15), (0, 16), (0, 17), (0, 18), (0,
19), (0, 20), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (1, 8), (1, 9),
(1, 10), (1, 11), (1, 12), (1, 13), (1, 14), (1, 15), (1, 16), (1, 17), (1, 18),
(1, 19), (1, 20), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (2,
10), (2, 11), (2, 12), (2, 13), (2, 14), (2, 15), (2, 16), (2, 17), (2, 18), (2,
19), (2, 20), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 9), (3, 10), (3, 11),
```

```
(3, 12), (3, 13), (3, 14), (3, 15), (3, 16), (3, 17), (3, 18), (3, 19), (3, 20),
(4, 5), (4, 6), (4, 7), (4, 8), (4, 9), (4, 10), (4, 11), (4, 12), (4, 13), (4,
14), (4, 15), (4, 16), (4, 17), (4, 18), (4, 19), (4, 20), (5, 6), (5, 7), (5,
8), (5, 9), (5, 10), (5, 11), (5, 12), (5, 13), (5, 14), (5, 15), (5, 16), (5,
17), (5, 18), (5, 19), (5, 20), (6, 7), (6, 8), (6, 9), (6, 10), (6, 11), (6,
12), (6, 13), (6, 14), (6, 15), (6, 16), (6, 17), (6, 18), (6, 19), (6, 20), (7,
8), (7, 9), (7, 10), (7, 11), (7, 12), (7, 13), (7, 14), (7, 15), (7, 16), (7,
17), (7, 18), (7, 19), (7, 20), (8, 9), (8, 10), (8, 11), (8, 12), (8, 13), (8,
14), (8, 15), (8, 16), (8, 17), (8, 18), (8, 19), (8, 20), (9, 10), (9, 11), (9,
12), (9, 13), (9, 14), (9, 15), (9, 16), (9, 17), (9, 18), (9, 19), (9, 20),
(10, 11), (10, 12), (10, 13), (10, 14), (10, 15), (10, 16), (10, 17), (10, 18),
(10, 19), (10, 20), (11, 12), (11, 13), (11, 14), (11, 15), (11, 16), (11, 17),
(11, 18), (11, 19), (11, 20), (12, 13), (12, 14), (12, 15), (12, 16), (12, 17),
(12, 18), (12, 19), (12, 20), (13, 14), (13, 15), (13, 16), (13, 17), (13, 18),
(13, 19), (13, 20), (14, 15), (14, 16), (14, 17), (14, 18), (14, 19), (14, 20),
(15, 16), (15, 17), (15, 18), (15, 19), (15, 20), (16, 17), (16, 18), (16, 19),
(16, 20), (17, 18), (17, 19), (17, 20), (18, 19), (18, 20), (19, 20)]
```

[160]:
```
x = [[1,2,3],[4,5,6],[7,8,9]]  # Flattening a 2d list into 1d list using A
 ↪single for loop and inbuilt function
k = []
[k.extend(i) for i in x]
```

[160]:
```
[None, None, None]
```

[161]:
```
k
```

[161]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

[145]:
```
p=[]
p.append([1,2])
p
```

[145]:
```
[[1, 2]]
```

[143]:
```
p
```

[ ]:
```

```