27/08/2020

17CP060

Tejas

Q.1)

⇒ A) string = 'CO2 H2O'
print(string.find('2'))

B) string = 'CO2 H2O'
x = 0
for i in range(2):
x = string.find('2', x)

Print(x)

C) string = 'Boolean'
if (s[0].islower())
print('True')

D) string = 'MoNdAy'
print(string.lower())

E) string = " Monday "
string.replace(" ", "") // string.strip()

F) string = 'boolean'
print(string.upper())

Q.2)

29/08/2020

17CP060

Jayeni.

```
temps_in_celsius = [#input]
temps_in_f = []

for i in temps_in_celsius :
    temps_in_f.append ((i * 9/5) + 32)

print (temps_in_f)
```

Q.3)

27/08/2020
17CP060
Tayyeb

```
n = int(input())

List1 = []

for i in range(n+1):
    if (i == 0):
        continue
    else:
        List1.append(i)
        print("(", List1, ")")

print("---------")

for i in range(n, -1, -1):
    if (i == 0):
        continue
    else:
        print(List1)
        print("(", List1, ")")

    if (i == 0):
        continue
    else:
        print List1.remove(List1[0])
```

Q.4.2) How lambda function is different from inline function?

21/08/2020
17CP060
Tejemi

⇒ lambda functions are inline function but they do not have name and to define them we have to use lambda keyword besides that lambda function have only one expression in their body.

Q.4.1) significance of else statement in the looping:

⇒ python provides facility of else with loop so we can use else block underneath loop body and it will execute after all iterations are over and if there, loop hasn't exited because break.

example:

```
for i in range(0,4):
    print(i)
else:
    print("else with loop")
```

Q.5) append and extend :

27/08/2020
17CP060
Tejaswi

⇒ the reason to have both append and extend is that append only adds 1 element to list while extend adds whole iterable (list) of argument to list.

when we use append where extend is supposed to use it adds whole list as one element.

Q.6) yes python utilize memory more efficiently compare to other languages cause it's use pool of memory to hold values of object and as soon as no object is reffering to that memory it is freed and given back to free memory pool. and it does not allocate extra memory to variables having same value.

for example x = 1

y = 1

here both x and y refer to same memory location

27/08/2020
17CP060
Jayeni

Q.6)

```
import random
l = []

for i in range(75):
    l.append(random.randint())

for i in l

file1 = open("odd.txt", "w")
file2 = open("even.txt", "w")

for i in l:
    if(i % l == 0):
        file2.write(str(i))
        Print("even",i)
    else:
        file1.write
        file1.write(str(i))
        Print("odd",i)
file1.close()
file2.close()
```