1. def valid\_rondo(in\_string):

in\_string\_x = in\_string.replace('A','')

in\_string\_x = list(in\_string\_x)

output = False

if len(in\_string\_x) == 1:

output= True

else:

for ele in range(len(in\_string\_x)-1):

if ord(in\_string\_x[ele]) < ord(in\_string\_x[ele+1]):

output= True

else:

output = False

break

print(f"valid\_rondo({in\_string}) ➞ {output}")

valid\_rondo("ABACADAEAFAGAHAIAJA")

valid\_rondo("ABA")

valid\_rondo("ABBACCA")

valid\_rondo("ACAC")

valid\_rondo("A")

Output:

valid\_rondo(ABACADAEAFAGAHAIAJA) ➞ True

valid\_rondo(ABA) ➞ True

valid\_rondo(ABBACCA) ➞ False

valid\_rondo(ACAC) ➞ False

valid\_rondo(A) ➞ False

1. def sentence\_searcher(in\_string, search\_text):

output = '""'

for ele in in\_string.split(". "):

if len(ele.lower().replace(search\_text.lower(),'')) != len(ele):

output = ele

break

print(f'sentence\_searcher{in\_string,search\_text} ➞ {output}')

txt = "I have a cat. I have a mat. Things are going swell."

sentence\_searcher(txt, "have")

sentence\_searcher(txt, "MAT")

sentence\_searcher(txt, "things")

sentence\_searcher(txt, "flat")

Output:

sentence\_searcher('I have a cat. I have a mat. Things are going swell.', 'have') ➞ I have a cat

sentence\_searcher('I have a cat. I have a mat. Things are going swell.', 'MAT') ➞ I have a mat

sentence\_searcher('I have a cat. I have a mat. Things are going swell.', 'things') ➞ Things are going swell.

sentence\_searcher('I have a cat. I have a mat. Things are going swell.', 'flat') ➞ ""

1. def sum\_round(in\_num):

output = []

in\_num = str(in\_num)

for ele in range(len(in\_num)):

if in\_num[ele] != '0':

output.append(in\_num[ele]+len(in\_num[ele+1:])\*'0')

print(f'sum\_round({in\_num}) ➞ {" ".join(output[::-1])}')

sum\_round(101)

sum\_round(1234)

sum\_round(54210)

Output:

sum\_round(101) ➞ 1 100

sum\_round(1234) ➞ 4 30 200 1000

sum\_round(54210) ➞ 10 200 4000 50000

1. def multiplication\_table(in\_num):

out\_list =[]

for a in range(1,in\_num+1):

temp\_list = []

for b in range(1,in\_num+1):

temp\_list.append(a\*b)

out\_list.append(temp\_list)

print(f'multiplication\_table({in\_num}) ➞ {out\_list}')

Output:

multiplication\_table(3) ➞ [[1, 2, 3], [2, 4, 6], [3, 6, 9]]

multiplication\_table(1) ➞ [[1]]

multiplication\_table(5) ➞ [[1, 2, 3, 4, 5], [2, 4, 6, 8, 10], [3, 6, 9, 12, 15], [4, 8, 12, 16, 20], [5, 10, 15, 20, 25]]

1. def does\_rhyme(in\_one,in\_two):

vowels = 'aeiou'

output= False

in\_one\_rhyme = [x.lower() for x in in\_one.split(" ")[-1] if x.lower() in vowels]

in\_two\_rhyme = [x.lower() for x in in\_two.split(" ")[-1] if x.lower() in vowels]

if in\_one\_rhyme == in\_two\_rhyme:

output = True

print(f'does\_rhyme{in\_one,in\_two} ➞ {output}')

does\_rhyme("Sam I am!", "Green eggs and ham.")

does\_rhyme("Sam I am!", "Green eggs and HAM.")

does\_rhyme("You are off to the races", "a splendid day.")

does\_rhyme("and frequently do?", "you gotta move.")

multiplication\_table(3)

multiplication\_table(1)

multiplication\_table(5)

Output:

does\_rhyme('Sam I am!', 'Green eggs and ham.') ➞ True

does\_rhyme('Sam I am!', 'Green eggs and HAM.') ➞ True

does\_rhyme('You are off to the races', 'a splendid day.') ➞ False

does\_rhyme('and frequently do?', 'you gotta move.') ➞ False