Strings2

```
# Quizzes

# + , *

name = "Rahul"
second = "Janghu"

print(ord('A'))

65

ord("Z")

90

ord("a")

97

ord("z")

122

name

'Rahul'

second

'Janghu'

name + second

'RahulJanghu'

name * 3

'RahulRahulRahul'
```

Pattern

```
# The cool way to print the pattern:

# # # # #
# # # # #
# # # # #
```

```
# # # # #
# # # # #

# Quiz

# Using nested for loop

for i in range(4): # This loop will run 4 times
    print("#", end=" ")

# # # #

for i in range(5):
    print("#", end=" ")
print()

for i in range(5):
    print("#", end=" ")
print()

for i in range(5):
    print("#", end=" ")
print()

for i in range(5):
    print("#", end=" ")
print()

for i in range(5):
    print("#", end=" ")

# # # # #
# # # # #
# # # # #
# # # # #
# # # # #



for i in range(5):
    for j in range(5): # This loop is printing 5 #
        print("#", end=" ")
    print()

# # # # #
# # # # #
# # # # #
# # # # #
# # # # #
```

```
# Using a single loop

# *, +

print("# " * 5)
print("# " * 5)
print("# " * 5)
print("# " * 5)
print("# " * 5)
```

```
# # # # #
# # # # #
# # # # #
# # # # #
# # # # #
```

```
for i in range(5):
    print("# " * 5)
```

```
# # # # #
# # # # #
# # # # #
# # # # #
# # # # #
```

Formatted strings

```
# intro please :)

name = "Rahul"
place = "Gurugram"

print("Hey my name is", name, "and i live in", place)
```

```
Hey my name is Rahul and i live in Gurugram
```

```
# 1st method

# format

# quiz

name = "Rahul"
place = "Gurugram"
```

```
print("Hey my name is {}. I live in {}.".format(name, place))
```

```
Hey my name is Rahul. I live in Gurugram.
```

```
name = "Rahul"
place = "Gurugram"

print("Hey my name is {}. I live in {}.".format(place, name))
```

```
Hey my name is Gurugram. I live in Rahul.
```

```
# latest way of writing formatted strings
# Following is aplicable in python 3.6 and above

# 2nd Method

# formatted strings

name = "Rahul"
place = "Gurugram"

print(f"Hey my name is {name}. I live in {place}.")
```

```
Hey my name is Rahul. I live in Gurugram.
```

```
name = "Rahul"
place = "Gurugram"

print("Hey my name is {name}. I live in {place}.")
```

```
Hey my name is {name}. I live in {place}.
```

```
# Quix

# name = 'Emma Watson'
# age = 32
# print("Hey my name is {}, and my age is {}.".format(age))
```

Challenge: For a given string find length without using len function

```
name = "Rahul Janghu"

len(name)
```

```
12
```

```
for i in name:
    print(i)
```

```
R
a
h
u
l

J
a
n
g
h
u
```

```
count = 0
```

```
for i in name:
    count += 1
    #print(count)
```

```
print(count)
```

```
12
```

Challenge:

- Given a string as input, count the no of upper case characters

```
ord("A")
```

```
65
```

```
ord("Z")
```

```
90
```

```
name = "RaHuL jAnghu"
```

```
for i in name:
    # Check for upper case
    if ord(i) >= 65 and ord(i) <= 90:
        print(i)
```

```
R
H
```

```
L
A
```

```
# Final code

count = 0

for i in name:
    # Check for upper case
    if ord(i) >= 65 and ord(i) <= 90:
        count += 1

print(count)
```

```
4
```

```
def counting(a):

    count = 0
    for i in a:
        # Check for upper case
        if ord(i) >= 65 and ord(i) <= 90:
            count += 1

    return count

counting(name)
```

```
4
```

```
# isupper() and upper()

"i".isupper()
```

```
False
```

```
"RAHUL".isupper()
```

```
True
```

```
"RaHul".isupper()
```

```
False
```

```
"RaHul".upper()
```

```
'RAHUL'
```

```
"RAHUL".lower()
```

```
'rahul'
```

```
weather = input()

if weather.lower() == "sunny":
    print("Yes")
```

```
 SUNNY
```

```
Yes
```

Challenge:

- Convert the string to lower case.
- Input: " RaHUl" Output: "rahul"

```
# using loop

name = input()
```

```
 RaHUl
```

```
name
```

```
'RaHUl'
```

```
# Upper and lower difference 32

new = ""
for i in name:
    if i.isupper():
        new += i.lower()
    else:
        new += i

print(new)
```

```
rahul
```

```
name[::-1]
```

```
'lUHaR'
```

```
# lower and islower

# using isupper and islower

name

'RaHUl'

name.islower()

False

name.lower()

'rahul'



# Convert upper to lower and lower to upper

# RaHuL -> rAhUl

name = "RaHuL"

for i in name:
    if i.isupper():
        print("upper", i)
    if i.islower():
        print("lower", i)

upper R
lower a
upper H
lower u
upper L

for i in name:
    if i.isupper():
        print(i.lower())
    if i.islower():
        print(i.upper())

r
A
h
U
l
```

```
# Make a new string out of it

new = ""
for i in name:
    if i.isupper():
        new += i.lower()
    if i.islower():
        new += i.upper()
```

```
name
```

```
'RaHuL'
```

```
new
```

```
'rAhUl'
```

```
# Try this question using ord
```

Challenge:

Write a program which accepts two strings s1 and s2 and checks if s2 is a substring of s1.

```
# in operator

# using for loop

name = "Rahul Janghu"

for i in name:
    if i == "a":
        print("Yes")
        break
```

```
Yes
```

```
if "a" in name:
    print("Yes")
else:
    print("NO")
```

```
Yes
```

```
if "aR" in name:
    print("Yes")
else:
    print("NO")
```

```
NO
```

```
name
```

```
'Rahul Janghu'
```

```
if "RaH" in name:
    print("Yes")
else:
    print("NO")
```

```
NO
```

```
if "Rahul Janghu" in name:
    print("Yes")
else:
    print("NO")
```

```
Yes
```

Challenge

Write the code for a Python function expand(x) that takes a list of strings, concatenates them, and returns the resulting string repeated three times.

Example 1:

Input: ['string1', 'string2']

Output: 'string1string2string1string2string1string2'

Example 2:

Input: ['a', 'b', 'c']

Output: 'abcabcabc'

```
# quiz
```

```
# split?
```

```
# using for loop
```

```
n = input().split()
```

```
 a b c
```

```
n
```

```
['a', 'b', 'c']
```

```
n = n * 3

res = ""
for i in n:
    print(i)
```

```
a
b
c
a
b
c
a
b
c
```

```
res = ""
for i in n:
    res += i

print(res)
```

```
abcabcabc
```

```
li = input().split()
```

```
 a b c
```

```
li
```

```
['a', 'b', 'c']
```

```
res = ""

for i in li:
    res += i

print(res)
```

```
abc
```

```
print(res * 3)
```

```
abcabcabc
```

```
# Join is basically concatinating strings through list
```

```
# "".join(li)
li = ["c", "b", "d", "e"]
"a".join(li)
'cabadae'
"-".join(li)
'c-b-d-e'
li = input().split()
 string1 string2
li
['string1', 'string2']
res = "".join(li)
res*3
'string1string2string1string2string1string2'


print("".join(input().split()) * 3)
 a b c
abcabcabc



# Quiz
s = "1 2 3 4"
print(s.split())
['1', '2', '3', '4']
print("Ra" in "rahul")
False
n = ['string1', 'string2']
print("Rahul".join(n))
string1Rahulstring2
"string1" + "Rahul" + "string2"
```

```
'string1Rahulstring2'


# isalpha
name
'Rahul Janghu'
"".isalpha()
False
name
'Rahul Janghu'
name.isalpha()
False
"Rahul".isalpha()
True


# isdigit()
"".isdigit()
False
name
'Rahul Janghu'
name.isdigit()
False
"2".isdigit()
True
```

```
# Quiz
```

```
print("2a".isdigit())
```

```
False
```

```
print("2".isdigit())
```

```
True
```

Challenge

- Given a string count number of digits in a string

```
s = "Rah1h2 7 h"

for i in s:
    if i.isdigit():
        print(i)
```

```
1
2
7
```

```
count = 0
for i in s:
    if i.isdigit():
        count += 1

print(count)
```

```
3
```

```
# isspace
```

```
"a".isspace()
```

```
False
```

```
" ".isspace()
```

```
True
```

```
" a".isspace()
```

```
False
```

```
# count: counts the number of substrings
```

```
"Rahul janghu"
```

```
'Rahul janghu'
```

```
"Rahul janghu".count("A")
```

```
0
```

```
"Rahul janghu".count("a")
```

```
2
```

```
"Rahul janghu".count("Ra")
```

```
1
```

```
# index

# It gives first occurence of a substring
# Gives error if substring is not present
```

```
name
```

```
'Rahul Janghu'
```

```
name.index("a")
```

```
1
```

```
name.index?
```

```
Docstring:
S.index(sub[, start[, end]]) -> int

Return the lowest index in S where substring sub is found,
such that sub is contained within S[start:end].  Optional
arguments start and end are interpreted as in slice notation.

Raises ValueError when the substring is not found.
Type:      builtin_function_or_method
```

```
"rahul.janghu@scaler.com".index("@")
```

```
12
```

```
# find

# Gives the first occurence
# If not present then returns -1
```

```
name
```

```
'Rahul Janghu'
```

```
name.find("A")
```

```
-1
```

```
name.index("a")
```

```
1
```

```
# alpha numeric: isalnum
```

```
name
```

```
'Rahul Janghu'
```

```
name.isalnum()
```

```
False
```

```
"1".isalnum()
```

```
True
```

```
"ad2".isalnum()
```

```
True
```

```
"as2@".isalnum()
```

```
False
```

```
# HW: Try the patterns using string concatenation

# More patterns?
```

#
##
###
####

####
###
##
#