String

1] escape sequence and raw string :

```
#naive solution

def isAnagram(s1,s2):
    if len(s1)!=len(s2):
        return False
    s1=sorted(s1)
    s2=sorted(s2)
    return (s1==s2)

s1="listen"
s2="silent"
print(isAnagram(s1,s2))

#efficient solution
print("**************** Efficient solution **************")
def isAnagram(s1,s2):
    if len(s1)!=len(s2):
        return False

    count=[0]*256
    for i in range(len(s1)):
        count[ord(s1[i])]+=1
        count[ord(s2[i])]-=1

    for x in count:
        if x!=0:
            return False
    return True

print(isAnagram(s1,s2))
```

OUTPUT :

```
True

**************** Efficient solution **************

True
```

2] Reverse string :

```
#string in python are immutable

#using loop
s=input("Enter a string: ")
rev=""
for i in s:
    rev=i+rev
print("reverse string is: ",rev)
print()


#using slicing
s=input("Enter a String: ")
print(s[::-1])

"""
OUTPUT
Enter a string: ganesh
reverse string is:  hsenag

Enter a String: ganesh
hsenag
"""
```

3] string comparison:

```
s1="geeksforgeeks"
s2="ide"

print(s1<s2)
print(s1<=s2)
print(s1>s2)
print(s1>=s2)
print(s1==s2)
print(s1!=s2)



print("##################")

print("abcd>abc","abcd">"abc")
print("ZAB>ABC","ZAB">"ABC")

print("abc>ABC","abc">"ABC")

print("x>abcd","x">"abcd")
```

OUTPUT :

```
True

True

False

False

False

True

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

abcd>abc True

ZAB>ABC True

abc>ABC True

x>abcd True
```

4] String in Python :

```
#ord and char
print(ord("a"))  #chr to ord   "unicode"
print(ord("A"))

print(chr(97))   # ord to chr    "unicode"
print(chr(65))
print()


#indexing

s="geek"
print(s)
print(s[0])
print(s[-1])
print(s[1])
print(s[-2])
print()


#multiline string
s="""Hi,
This is "python course"
Hope you are enjoying it."""
print(s)
print()



#string are immutable
s = "geek"
s[0] = "e"  # error: item assignment not supported
print(s)
```

OUTPUT :

```
Traceback (most recent call last):
  File "E:/programming/DSA in Python/String/4_string_in_python.py", line 32, in <module>
    s[0] = "e"  # error: item assignment not supported
TypeError: 'str' object does not support item assignment
```

```
97
65
a
A

geek
g
k
e
e

Hi,
This is "python course"
Hope you are enjoying it.
```

5] Formated string in python :

```
#3 formated
name="ABC"
course="Python Course"

s="Welcome %s to the %s"%(name,course)
print(s)
print()

# using format function
s="Welcome {0} to the {1}".format(name,course)
print(s)
print()

#using f-string
s=f"welcome{name} to the {course}"
print(s)
print()


print("#######################################")

# f-string
a=10
b=20

print(f"sum of {a} and {b} is {a+b}")
print(f"product of {a} and {b} is {a*b}")


print("#######################################")
s1="ABC"
s2="abc"

print(f"lower case of {s1} is {s1.lower()}")
print(f"upper case of {s2} is {s2.upper()}")
```

OUTPUT :

Welcome ABC to the Python Course

Welcome ABC to the Python Course

```
welcomeABC to the Python Course

###################################
sum of 10 and 20 is 30
product of 10 and 20 is 200
###################################
lower case of ABC is abc
upper case of abc is ABC
```

6] String Operation :

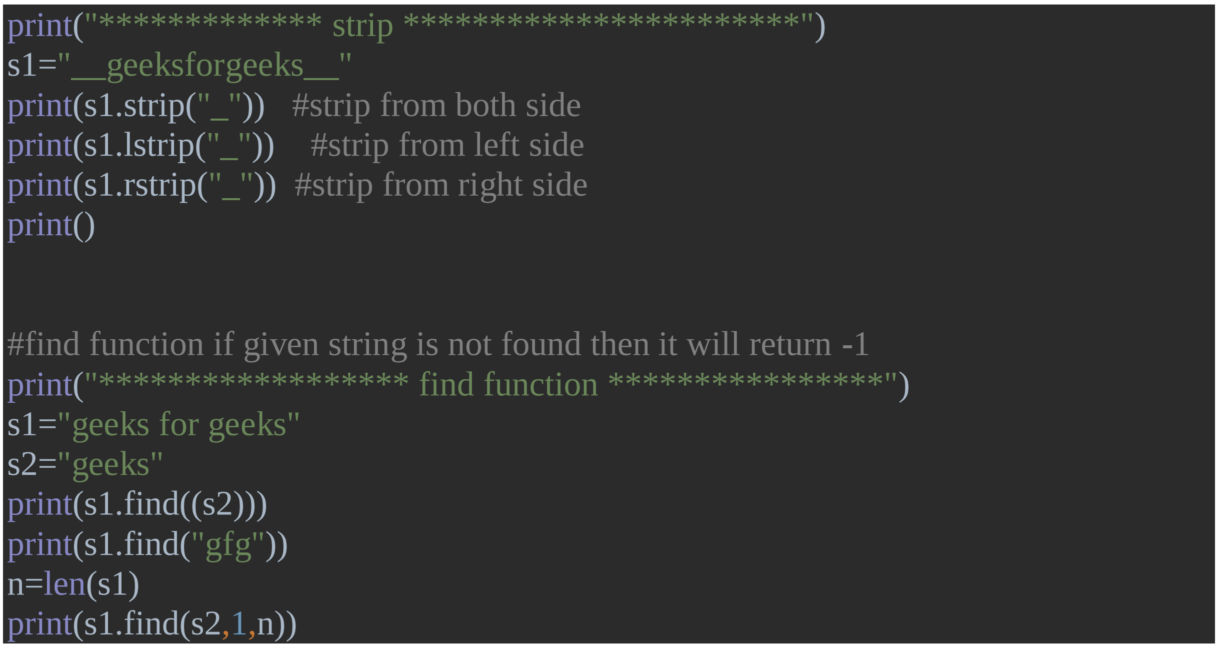
```
#checking for substring
s1="geeksforgeeks"
s2="geeks"

print(s2 in s1)
print( s2 not in s1)

#concatenation
s1="geeks"
s2="for"
s3=s1+s2

s4="welcome to "+ s1 + s2

print(s3)
print(s4)


#Position of Substring
s1="geeksforgeeks"
s2="geek"

print(s1.index(s2))
print(s1.rindex(s2))  #right index
print(s1.index(s2,0,13))  #start and end index
```

OUTPUT :

```
True

False

geeksfor

welcome to geeksfor

0

8

0
```

7] String Operation Part 2:

```
#len, lower and upper
print("************ Len, lower and upper ************** ")
s1="geeks"

print(len(s1))

s2=s1.upper()

print(s2)

s3=s2.lower()
print(s3)

print(s1.islower())
print(s2.isupper())
print()


#startwith and endwith function
print("******** startwith and endwith function ***********")
s="GeeksforGeeks Python Course"

print(s.startswith("Geeks"))
print(s.endswith("Course"))
print(s.startswith("Geeks",1))  #start index
print(s.startswith("Geeks",8,len(s)))  #start index, last index
print()

#split and join
#split function convert string into list and viceversa
print("********* split and join funciton *****************")
s1="geeks for geeks"
print(s1.split())      #split by space

s2="geeks, for, geeks"
print(s2.split(','))   #split by comma

l=["geeksforgeeks","python","course"]
print(" ".join(l))      #join by space
print(",".join(l))      #join by comma


#strip function remove the unwanted character from the sting
```



```
print("************* strip ***********************")
s1="__geeksforgeeks__"
print(s1.strip("_"))   #strip from both side
print(s1.lstrip("_"))    #strip from left side
print(s1.rstrip("_"))  #strip from right side
print()


#find function if given string is not found then it will return -1
print("****************** find function ****************")
s1="geeks for geeks"
s2="geeks"
print(s1.find((s2)))
print(s1.find("gfg"))
n=len(s1)
print(s1.find(s2,1,n))
```

OUTPUT :

```
************ Len, lower and upper **************

5

GEEKS

geeks

True

True


******** startwith and endwith function ***********

True

True

False

True


********* split and join funciton ****************

['geeks', 'for', 'geeks']
```

```
['geeks', ' for', ' geeks']

geeksforgeeks python course

geeksforgeeks,python,course

************* strip ***********************

geeksforgeeks

geeksforgeeks__

__geeksforgeeks


****************** find function ****************

0

-1

10
```

8] Pattern Searching in Python:

```
txt=input("Enter Text:\n")
pat=input("Enter Pattern:\n")

pos=txt.find(pat)

while pos>=0:
    print(pos)
    pos=txt.find(pat,pos+1)


"""
OUTPUT

Enter Text:
geeks for geeks
Enter Pattern:
geeks
0
10
"""
```

9] Check for Palindrome :

```
#using while loop
print("********* Using While Loop *********************")
s=input("Enter a string:\n")
low=0

high=len(s)-1

while low<high:
    if s[low]!=s[high]:
        print("No, Given string is not palindrom")
        break
    low+=1
    high-=1

else:
    print("yes, Given string is palindrome")

print()

#using slicing
print("************* Using Slicing **********************")

if s==s[::-1]:
    print("Yes, Given string is Palindrome")
else:
    print("No, Given string is not palindrome")
```

OUTPUT :

```
1] ********* Using While Loop *********************

Enter a string:

 ganesh

No, Given string is not palindrom


************* Using Slicing **********************

No, Given string is not palindrome
```

```
2] ********* Using While Loop *********************

Enter a string:

geeg

yes, Given string is palindrome


************* Using Slicing **********************

Yes, Given string is Palindrome
```

10] For anagram :

```
#naive solution

def isAnagram(s1,s2):
    if len(s1)!=len(s2):
        return False
    s1=sorted(s1)
    s2=sorted(s2)
    return (s1==s2)

s1="listen"
s2="silent"
print(isAnagram(s1,s2))


#efficient solution
print("**************** Efficient solution **************")
def isAnagram(s1,s2):
    if len(s1)!=len(s2):
        return False

    count=[0]*256
    for i in range(len(s1)):
        count[ord(s1[i])]+=1
        count[ord(s2[i])]-=1

    for x in count:
        if x!=0:
            return False
    return True


print(isAnagram(s1,s2))
```

OUTPUT :

```
True

**************** Efficient solution **************

True
```