```
# In Python, a string is an immutable sequence of Unicode characters. Each character has a unique numeric value as per the UNIC
# But, the sequence as a whole, doesn't have any numeric value even if all the characters are digits.
# To differentiate the string from numbers and other identifiers, the sequence of characters is included within single, double
# Hence, 1234 is a number (integer) but '1234' is a string.
```

```
s1 = 'Govind'
s2 = "Rama"
s3 = """Parab"""
s4 = '''Govind Rama Parab'''
s5 = '''Govind
is
my
name'''
print(s1,s2,s3,s4,s5)
```

```
Govind Rama Parab Govind Rama Parab Govind
is
my
name
```

```
# accessing string and slicing
print(s1[::])
print(s1[::-1])
print(s1[:2])
print(s1[2:])
print(s1[2])
print(s1[-3:])
print(s1[:-3])
print(s1[0:1])
```

```
Govind
dnivoG
Go
vind
v
ind
Gov
G
```

```
# updating string
s1 = s1+" Parab"
print(s1)

s4 = s4[:11]+" Parab 1"
print(s4)
```

```
Govind Parab
Govind Rama Parab 1
```

```
# format in string
var = 20
s1 = f"This is for {var} dollars"
print(s1)
```

```
This is for 20 dollars
```

```
# membership operator
print("d" in s1)
print("z" not in s1)
print("d" not in s1)
print("z" in s1)
```

```
True
True
False
False
```

```
# repetition
s1 = "Ha"*5
print(s1)
```

```
HaHaHaHaHa
```

```
# type
print(type(s1))
```

```
<class 'str'>
```

```
# multi-line string
var = """
this is
my name
govind
"""
print(var)
```

```

this is
my name
govind
```

```
# adding quotes to string
var = 'Welcome to "Python Tutorial" from Colab'
print ("var:", var)

var = "Welcome to 'Python Tutorial' from Colab"
print ("var:", var)
```

```
var: Welcome to "Python Tutorial" from Colab
var: Welcome to 'Python Tutorial' from Colab
```

```
# String Formatting Operator
print ("My name is %s and weight is %d kg!" % ('Zara', 21))

str1 = "Welcome to {}"
print(str1.format("Donland"))
```

```
My name is Zara and weight is 21 kg!
Welcome to Donland
```

```
# modifying string
s1="WORD"
print ("original string:", s1)
l1=list(s1)
l1.insert(0,"My")
l1.insert(4,"L")

print (l1)

s1=''.join(l1)
print ("Modified string:", s1)
```

```
original string: WORD
['My', 'W', 'O', 'R', 'L', 'D']
Modified string: MyWORLD
```

```
# Concatenation
str1="Hello"
str2="World"
print ("String 1:",str1)
print ("String 2:",str2)
str3=str1+str2
print("String 3:",str3)
```

```
String 1: Hello
String 2: World
String 3: HelloWorld
```

```
# basic methods
s1 = "Hello Govind rama parab"
print(s1.capitalize())#Capitalizes first letter of string
print(s1.title())
print(s1.lower())
print(s1.upper())
print(len(s1))
print(s1.swapcase())
```

```
Hello govind rama parab
Hello Govind Rama Parab
hello govind rama parab
HELLO GOVIND RAMA PARAB
23
hELLO gOVIND RAMA PARAB
```

```
# basic alignment methods

# center(width, fillchar)
# Centers the string, padding it with fillchar (default is space) to make the total width equal to width.
print('hello'.center(11, '*'))  # Output: '***hello***'

# ljust(width[, fillchar])
# Left-aligns the string, padding it on the right with fillchar (default is space).
print('hello'.ljust(10, '-'))  # Output: 'hello-----'

#rjust
print('hello'.rjust(10, '-'))  # Output: '-----hello'

# zfill(width)
# Pads the string on the left with zeros (0) to make the total width equal to width. If the string has a sign (+ or -), it is p

print('-42'.zfill(5))  # Output: '-0042'

# expandtabs(tabsize=8)
# Replaces tab characters (\t) with spaces. Default tab size is 8 unless specified.

print('hello\tworld'.expandtabs(4))  # Output: 'hello   world'
```

```
***hello***
hello-----
-----hello
-0042
hello   world
```

```
# split and join methods

text = "   Hello, Python World!   \nWelcome to string methods.\n"
text_with_tabs = "\tTabbed\ttext"
long_text = "one-two-three-two-one"
sequence = ['Learn', 'Python', 'Now']
prefixed = "Mr.John, Mr.Govind"
suffixed = "report.pdf, result.pdf"

# 1. lstrip()
print("1. lstrip():", repr(text.lstrip()))

# 2. rstrip()
print("2. rstrip():", repr(text.rstrip()))

# 3. strip()
print("3. strip():", repr(text.strip()))

# 4. rsplit()
print("4. rsplit():", long_text.rsplit('-', 2))  # splits from right, max 2 splits

# 5. split()
print("5. split():", text.strip().split())  # default is whitespace

# 6. splitlines()
print("6. splitlines():", text.splitlines())

# 7. partition()
print("7. partition():", long_text.partition('-'))  # splits at first '-'

# 8. rpartition()
print("8. rpartition():", long_text.rpartition('-'))  # splits at last '-'

# 9. join()
print("9. join():", ' '.join(sequence))

# 10. removeprefix()
print("10. removeprefix():", prefixed.removeprefix("Mr."))

# 11. removesuffix()
print("11. removesuffix():", suffixed.removesuffix(".pdf"))
```

```
1. lstrip(): 'Hello, Python World!   \nWelcome to string methods.\n'
2. rstrip(): '   Hello, Python World!   \nWelcome to string methods.'
3. strip(): 'Hello, Python World!   \nWelcome to string methods.'
4. rsplit(): ['one-two-three', 'two', 'one']
5. split(): ['Hello,', 'Python', 'World!', 'Welcome', 'to', 'string', 'methods.']
6. splitlines(): ['   Hello, Python World!   ', 'Welcome to string methods.']
7. partition(): ('one', '-', 'two-three-two-one')
8. rpartition(): ('one-two-three-two', '-', 'one')
9. join(): Learn Python Now
10. removeprefix(): John, Mr.Govind
11. removesuffix(): report.pdf, result
```

```
# Boolean methods

str1 = "Python3"
str2 = "Python"
str3 = "12345"
str4 = "hello"
str5 = "Ⅻ"           # Roman numeral twelve – numeric but not digit
str6 = "   "
str7 = "Hello World"
str8 = "HELLO"
str9 = "abc123"
str10 = "π"           # Greek pi character, not ASCII
str11 = "variable_name"
str12 = "This is printable!"

# 1. isalnum()
print("1. isalnum():", str1.isalnum())  # True

# 2. isalpha()
print("2. isalpha():", str2.isalpha())  # True

# 3. isdigit()
print("3. isdigit():", str3.isdigit())  # True

# 4. islower()
print("4. islower():", str4.islower())  # True

# 5. isnumeric()
print("5. isnumeric():", str5.isnumeric())  # True

# 6. isspace()
print("6. isspace():", str6.isspace())  # True

# 7. istitle()
print("7. istitle():", str7.istitle())  # True

# 8. isupper()
print("8. isupper():", str8.isupper())  # True

# 9. isascii()
print("9. isascii():", str9.isascii())  # True

# 10. isdecimal()
print("10. isdecimal():", str3.isdecimal())  # True

# 11. isidentifier()
print("11. isidentifier():", str11.isidentifier())  # True

# 12. isprintable()
print("12. isprintable():", str12.isprintable())  # True

# isprintable() is a string method used to check whether all characters in a string are printable or not.

# It returns:

# True → if every character can be displayed (letters, numbers, symbols, space)

# False → if the string contains non-printable characters (like newline \n, tab \t, backspace, etc.)
```

```
1. isalnum(): True
2. isalpha(): True
3. isdigit(): True
4. islower(): True
5. isnumeric(): True
6. isspace(): True
```

```
7. istitle(): True
8. isupper(): True
9. isascii(): True
10. isdecimal(): True
11. isidentifier(): True
12. isprintable(): True
```

```
# Find and replace methods

text = "Python is easy. Python is powerful. Python is popular."

# 1. count(sub, beg, end)
print("1. count('Python'):", text.count('Python'))  # Output: 3

# 2. find(sub, beg, end)
print("2. find('Python'):", text.find('Python'))  # Output: index of 'powerful'

# 3. index(sub, beg, end)
print("3. index('easy'):", text.index('easy'))  # Output: index of 'easy'

# 4. replace(old, new [, max])
print("4. replace('Python', 'Java'):", text.replace('Python', 'Java'))  # Replace all
print("   replace('Python', 'Java', 2):", text.replace('Python', 'Java', 2))  # Replace first 2

# 5. rfind(sub, beg, end)
print("5. rfind('Python'):", text.rfind('Python'))  # Finds last occurrence

# 6. rindex(sub, beg, end)
print("6. rindex('Python'):", text.rindex('Python'))  # Last index of 'Python'

# 7. startswith(sub, beg, end)
print("7. startswith('Python'):", text.startswith('Python'))  # True

# 8. endswith(suffix, beg, end)
print("8. endswith('popular.'):", text.endswith('popular.'))  # True
```

```
1. count('Python'): 3
2. find('Python'): 0
3. index('easy'): 10
4. replace('Python', 'Java'): Java is easy. Java is powerful. Java is popular.
   replace('Python', 'Java', 2): Java is easy. Java is powerful. Python is popular.
5. rfind('Python'): 36
6. rindex('Python'): 36
7. startswith('Python'): True
8. endswith('popular.'): True
```

```
# Transalte functions

# Original string
text = "hello world 123"

# 1. maketrans()
# Replace h->H, e->3, l->1, o->0 and remove '1', '2'
trans_table = str.maketrans({'h': 'H', 'e': '3', 'l': '1', 'o': '0', '1': None, '2': None})

# 2. translate()
translated_text = text.translate(trans_table)

print("Original text:   ", text)
print("Translated text: ", translated_text)
```

```
Original text:    hello world 123
Translated text:  H3110 w0r1d 3
```

```
# escape characters

print("1. Newline (\\n):\nThis text appears on a new line.")
print("2. Tab (\\t):\tThis text is tab-indented.")
print("3. Backslash (\\\\): This is a backslash: \\")
print("4. Single Quote (\\'):", 'It\'s a Python example.')
print("5. Double Quote (\\\"):", "He said, \"Hello!\"")
print("6. Bell (\\a): Triggers a system sound (may not work in all terminals)\a")
print("7. Backspace (\\b): Text\b\b\b123")  # May delete characters visually
print("8. Form Feed (\\f): First\fSecond")  # Rarely used, acts like page break
print("9. Carriage Return (\\r): Overwrites\rStart")  # Overwrites 'Overwrites'
print("10. Vertical Tab (\\v): Line1\vLine2")  # Rare use, vertical spacing
print("11. Hex value (\\xhh):", "\x48\x69")  # Hex for 'Hi'
```

```
print("12. Unicode (\\u):", "\u2764")  # Unicode heart ❤
print("13. Unicode (\\U):", "\U0001F600")  # Unicode smiley 😀
print("14. Octal value (\\ooo):", "\110\151")  # Octal for 'Hi'
```

```
1. Newline (\n):
This text appears on a new line.
2. Tab (\t):    This text is tab-indented.
3. Backslash (\\): This is a backslash: \
4. Single Quote (\'): It's a Python example.
5. Double Quote (\"): He said, "Hello!"
6. Bell (\a): Triggers a system sound (may not work in all terminals)
7. Backspace (\b): T123
8. Form Feed (\f): FirstSecond
Start
10. Vertical Tab (\v): Line1Line2
11. Hex value (\xhh): Hi
12. Unicode (\u): ❤
13. Unicode (\U): 😀
14. Octal value (\ooo): Hi
```

```
# ignore \
s = 'This string will not include \
backslashes or newline characters.'
print (s)

# escape backslash
s=s = 'The \\character is called backslash'
print (s)

# escape single quote
s='Hello \'Python\''
print (s)

# escape double quote
s="Hello \"Python\""
print (s)

# escape \b to generate ASCII backspace
s='Hel\blo'
print (s)

# ASCII Bell character
s='Hello\a'
print (s)

# newline
s='Hello\nPython'
print (s)

# Horizontal tab
s='Hello\tPython'
print (s)

# form feed
s= "hello\fworld"
print (s)

# Octal notation
s="\101"
print(s)

# Hexadecimal notation
s="\x41"
print (s)
```

```
This string will not include backslashes or newline characters.
The \character is called backslash
Hello 'Python'
Hello "Python"
Helo
Hello
Hello
Python
Hello   Python
helloworld
A
A
```