```
# Entire phrase

phrase = 'Statistics sits at the heart of machine learning'

print (phrase)
```

```
Statistics sits at the heart of machine learning
```

```
# Statement to get the type of the variable

type (phrase)
```

```
str
```

```
# We can also use double quote

my_string = "String built with double quotes"

print(my_string) #Use the print command
```

```
String built with double quotes
```

```
#Be careful with quotes!

sentence= 'I\'m using single quotes, but this will create an error'

print(sentence)
```

```
I'm using single quotes, but this will create an error
```

```
sentence= "I\"m using single quotes, but this will create an error"

print(sentence)
```

```
I"m using single quotes, but this will create an error
```

```
hashtag = "#"

print(hashtag)
```

```
#
```

```
type(hashtag)
```

```
str
```

```
#print('Linear Algebra')

#print('Calculus')

print('Use to print a new line')
```

```
print('\nSee what I mean?')
```

```
Use to print a new line

    See what I mean?
```

```
algo = 're on'
```

```
len(algo)
```

```
5
```

```
#Assign string as a string
```

```
string = 'Principal Component Analysis!'
```

```
#Assign string as a string
```

```
string = 'Principal Component Analysis!'
```

```
#Print the object
```

```
print(string)
```

```
Principal Component Analysis!
```

```
# Show first element (in this case a letter)
```

```
print(string[-2])
```

```
s
```

```
print(string[15])
```

```
n
```

```
len(string)
```

```
29
```

```
# Show first element (in this case a letter)
```

```
print(string[-2])
```

```
s
```

```
#Grab the element at the index 1, which is the LAST element
```

```
print(string[28])
```

```
!
```

```
print(string[-2])
```

```
s
```

```
# Grab everything past the first term all the way to the length of s which is len(s)
```

```
print(string)
```

```
print(string[1:])
```

```
Principal Component Analysis!
rincipal Component Analysis!
```

```
string[13]
```

```
'p'
```

```
string[12]
```

```
'm'
```

```
# Grab everything starting from index 10 till index 18
```

```
print(string[10:])
```

```
Component Analysis!
```

```
print(string[3:5])
```

```
nc
```

```
print(string[2:4])
```

```
in
```

```
#Last letter (one index behind so it loops back around)
```

```
string[-2]
```

```
's'
```

```
#Everything
```

```
print(string[:])
```

```
print(string)
```

```
Principal Component Analysis!
Principal Component Analysis!
```

```
#Grab everything, but go in steps size of 1

print(string)

print(string[::3])
```

```
Principal Component Analysis!
Pnp mntnys
```

```
print(string[::3])
```

```
Pnp mntnys
```

```
# Grab everything, but go in step sizes of 5

print(string)

print(string[5:15:5])
```

```
Principal Component Analysis!
iC
```

```
string[::1]
```

```
'Principal Component Analysis!'
```

```
#We can use this to print a string backwards

print(string)

string[::-1]
```

```
Principal Component Analysis!
'!sisylanA tnenopmoC lapicnirP'
```

```
print(string)

string[2:4:-1]
```

```
Principal Component Analysis!
''
```

```
#We can use this to print a string backwards with steps

print(string)

string[2:4:-1]
```

```
Principal Component Analysis!
    ''
```

```
string [4:2:-1]
```

```
'cn'
```

```
s='foobar'
```

```
s[0::-3]
```

```
'f'
```

```
# Concatenate strings!
```

```
string1='abc'
```

```
string2='def'
```

```
print(string1 + string2)
```

```
abcdef
```

```
#Concatenate strings!
```

```
string1='abc'
```

```
string2='def'
```

```
num = 4
```

```
print(string1 + str(4) + string2)
```

```
abc4def
```

```
print(string1 +' 4'+ string2)
```

```
abc 4def
```

```
str(num)
```

```
'4'
```

```
#Concatenate strings!
```

```
string1='abc'
```

```
string2='def'
```

```
string1 + '4'+ string2
```

```
'abc4def'
```

```
#We can reassign string completely though!

string = string + 'concatenate me!'

print(string)
```

```
Principal Component Analysis!concatenate me!
```

```
letters = 'wubba'

letters*2
```

```
'wubbawubba'
```

```
algorithm ='Neural Networks'

print(algorithm)
```

```
Neural Networks
```

```
#Print the length of the string

len(algorithm)
```

```
15
```

```
print(algorithm)
```

```
Neural Networks
```

```
algorithm.count('Networks')
```

```
1
```

```
algorithm.count('eu')
```

```
1
```

```
algorithm.count('')
```

```
16
```

```
algorithm.count('Neural')
```

```
1
```

```
algorithm.count('Neurla')
```

```
0
```

```
print(algorithm)
```

```
Neural Networks
```

```
algorithm.find('r')
```

```
3
```

```
algorithm.find('Neural')
```

```
0
```

```
algorithm.find('Box')
```

```
-1
```

```
algorithm.replace('','')
```

```
'Neural Networks'
```

```
algorithm.replace('N', 'L')
```

```
'Leural Letworks'
```

```
#Storing the modified string

algorithm_revised =algorithm.replace('Neural', 'Artificial Neural')

print(algorithm_revised)

print(algorithm)
```

```
Artificial Neural Networks
Neural Networks
```

```
first_name = 'Rahul'

last_name = 'Modi'

full_name = f'Left plus right makes (last_name) (first_name)'

print(first_name +last_name)
```

```
RahulModi
```

```
_name = 'Vikash'

e_name = ' '

name ='Srivastava'
```

```
name = f'I am none other than {first_name} {middle_name}{last_name}. I am a Data Scientist'

(full_name)
```

```
I am none other than Vikash  Srivastava. I am a Data Scientist
```

```
print(f'I am none other than{first_name} {middle_name}{last_name}. I am a Data Scientist')
```

```
I am none other thanVikash  Srivastava. I am a Data Scientist
```

```
my_string='Albert Einstein'

'Thomson' in my_string
```

```
False
```

```
'Alberta' in my_string
```

```
False
```