**PRINTING FORMATING**

*# .format() method  
# f-strings method*print('this is a string {}'.format('Inserted'))  
print("The {} {} {}".format('fox','brown','quick'))  
print("The {2} {0} {1}".format('fox','brown','quick'))  
print("The {0} {0} {0}".format('fox','brown','quick'))  
print("The {f} {b} {q}".format(f='fox',b='brown',q='quick'))  
  
result=100/777  
*# float formating follows {value:width.precision f}*print("The result was {r:10.3f}".format(r=result))  
  
*# string literal method*name="Jose"  
print(f'Hello, his name is {name}')  
  
  
*# Basic formating*print("Basic Formating")  
print('Old way-- %s %s'%('one','two'))  
print('New way method1-- {} {}'.format('one','two'))  
a='one'  
b='two'  
print(f'New way method2 string method-- {a} {b}')  
  
*# padding and aliggning strings*'''  
>By default values are formateed to take up only as many charcters as needed to represent the content  
>It is however also possible to define that a value shiuld be padded to a specific length   
>Unfortunately the default alignment differs between old and new style formatting  
>The old style defaults to right aligned while for news style it's left  
'''  
*# Align right*print('old %10s'%('test'))  
print('new {:>10}'.format('test'))  
*# Align left*print("old %-10s shiva"%('test'))  
print('new {:10} shiva'.format('test'))  
*# you can choose padding charcter*print('{a:\_>10}'.format(a='test'))  
print('{a:\_<10}'.format(a='test'))  
*# print('{a:\_10}'.format(a='test')) wrong formate  
  
# Center align Values*print('{:\_^10}'.format('test'))  
print('{:\_^10}'.format('zip'))  
  
*# Truncating long strings*'''  
>Inverse to padding it is also possible to truncate overly long values to a specific number of charcters.  
>The number behind a . in the format specifies the precision of the output .For strings  
that means that the output is truncated to the specified length . In our example this would  
be 5 charcters.  
'''  
print('old %.5s'%('xylophone'))  
print('New {:\_<10.5}'.format('xylophone'))  
  
print('New {:.5}'.format('xylophone'))  
  
*# Combining truncating and padding*print('Old %-10.5s'.format('xylophone'))  
print('New {:\_<10.5}'.format('xylophone'))  
  
*# Numbers  
# Integers*print("{}".format(43))  
  
print('Old %d'%(42))  
print('{:d}'.format(42))  
  
*# floats:*print("%f"%(3.141592653589793))  
print('{:f}'.format(3.14592653589793))  
  
*# Padding numbers*print('%4d'%(42))  
print('{:4d}'.format(42))  
print('{:\_>4d}'.format(42))  
print('{:\_<4d}'.format(42))  
  
*# floating point*print('%06.d'%(3.141592653589793))  
print('{:06.2f}'.format(3.141592653589793))  
print('{:\_>6.2f}'.format(3.141592653589793))  
  
*# signed numbers*print('%+d'%(42))  
print('{:>+10d}'.format(42))  
  
print('{:=5d}'.format(-23))  
print('{:=+5d}'.format(23))  
  
*# placeholder*print('{first} {last}'.format(first='shiva' ,last='srivastava'))  
  
*# Datetime*from datetime import datetime  
print('{:%Y-%m-%d %H:%M}'.format(datetime(2022,5,10,2,17)))  
'''  
from datetime import datetime  
dt = datetime(2001, 2, 3, 4, 5)  
New  
'{:{dfmt} {tfmt}}'.format(dt, dfmt='%Y-%m-%d', tfmt='%H:%M')  
Output  
2001-02-03 04:05  
  
'''