Day 14

```
def hello():
              pri nt( " hello world")
invoke it with hello()
def welcom e(n ame):
                pri nt( f'H ello, {name} ")
invoke it with welcom ("am y"
def welcom e_g ree tin g(name, greeting)
                pri nt(f' Hey{name}. {greet -
ing }")
invoke it with welcom e_g ree tin g(' Liz', "How
are you?") --> Kwargs
def expone nt( bas e,e xpo nent):
              power = base ** exponent
              return power
num1 = 2
num2 = 3
answer = expone nt( num 1,num2)
print( answer)
print( exp one nt( 2,3))
def sum(*p ara met ers): --> unlimited parameters
                total = 0
for i in parame ters:
                total = total + each_n umber
                return total
sum = sum(1, 2,3 ,4,5)
print(sum)
```

Day 16

```
Df = pd.read_csv(url)
print( df.t o_ str ing())
Null is null, a null is something = a special
creation to indicate the absence of a value - its a
made up value
Df.shape = not a method does not need ()
Look at only one column = df[‘School Name’]
```

Day 16 (cont)

> print(df[‘School Name’].to_string( ))
Find the unique names= df[‘School Name’].unique()
type(unique_schools) shows the type.. This is not a data frame
Statistics:
df[‘Starting Salary’].max() or df[‘Starting Salary’].mean() or df[‘St-arting Salary’].min()
Find the NAs = df[‘Starting Salary’].isna() then to count the trues = na_rows.sum()
Based on a condition
Df2 = df.query(“ `Starting Salary` > 75000 “)

Day 20

```
Change individual values
Df.loc[20, ‘Starting Salary’] = ‘ ‘
Convert to numeric type = df[‘St arting Salary’] =
pd.to_ num eric( df[‘St arting Salary’] )
Df.loc [139, ‘Starting Salary’] = 46000
#Convert starting salary to numeric FORCE CONVERT
or “COERCE” conversion
Error_ columns = pd.to_ num eric( df[‘St arting
Salary’], errors= ‘coerce’)
print( err or_ col umns)
#find the NAs
Nas = error_ col umn s.i sna()
print(Nas)
Df[20:25]
#fix columns
Df.loc[70, ‘Starting Salary’] = 42600
df[Nas]
Save it to the original by overwr iting
df[‘St arting Salary’] = pd.to_ num eri c(d f[‘ -
Sta rting Salary’])
```

Day 15

```
import pandas as pd
data_list = [45,74,78]
series _of _nu mbers = pd.ser ies (da ta_ list)
print( ser tie s_o f_n umb ers[1]) --> 74
Years = [2021, 2022, 2023]
series _of _nu mbers = pd.ser ies (da ta= dat -
a_list, index= years)
print( ser ies _of _nu mbers)
print( ser ies _of _nu mbe rs[ 2021] --> 2021
grades = {'A': 34, 'B': 56}
grade_ series = pd.ser ies (da ta= grades)
print( gra de_ series) or print( gra de_ ser ies -
['A']
quiz_s cores = {
                          'Qu iz1': [32,56 ,56],
                          'Qu iz2': [78,34 ,32]}
df = pd.Dat aFr ame (da ta= qui z_s cores)
print(df)
# to overwrite -->
df = pd.Dat aFr ame (da ta= qui z_s cores. index= -
['m ike', 'susan', 'amy']
df.head() --> top 5 rows
df.tail() --> bottom 5 rows
df[40:60] --> rows 40-59
```

Day 18

```
Find all the schools with the name Pitt
Df2 = df.query( " School Name == 'Pitt' ")
df2.head()
Remove a column: df.dro p(c olu mns ='S tarting
Salary', inplace = True)
Or df = df.dro p(c olu mns ='S tarting Salary')
Drop a row df.dro p(i ndex=2, inplace = True)
Delete entire row of data when one column had
missing data df=df.d ro pna()
```

Day 19

```
Load descriptives for the df = df.describe()
Load tab-de lim itted file
Df2 = pd.rea d_c sv(URL, sep='\t' )
Replace function:
Df[ 'School Name'].re pla ce('-', ' -', regex= -
True, inplac e=True)
Fillnas = df['St arting Salary '].f il lna(0,
inplac e=True)
How many unique school names are there:
len( df['School Name'].un ique())
Show only the rows in which df are duplicate:
Duplicates = df.dup lic ate d(s ubset= 'School
Name')
Boolean series = df[dup lic ates]
Df2 = df.dro p_d upl ica tes (su bse t=' School
Name', keep=' first')
Find out schools with specified
PA_schools = df2['S chool Name'].st r.c ont ain -
s(' Pen nsy lva nia')
Use a boolean series df2[PA _sc hools]
Overwrite instead on inplace
Df2 = df2.so rt_ val ues ('S tarting Salary',
ascending = False)
Fix one bad value:
Df2.loc[2, 'Starting Salary'] = df2['S tarting
Salary '].m ean()
```