In [22]:

```
# 01 - Crie um FUNÇÃO que mprema a sequenca de números PARES entre 1 a 30,
# e depois faça a chamada a função para listar apenas os numeros pares

def listaPar():
    for i in range(2,30,2):
        print(i)

listaPar()
```

```
2
4
6
8
10
12
14
16
18
20
22
24
26
28
```

In [35]:

```
# 02 - Crie um FUNÇÃO que mprema a sequenca de números IMPARES entre 1 a 30,
# e depois faça a chamada a função para listar apenas os numeros pares

def listaImpar():
    for i in range(1,30,2):
        print (i)
listaImpar()
```

```
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
```

In [31]:

```
# FUNÇÃO
def nome ( ):
    print('Edison Neves Soares')
nome()
```

```
Edison Neves Soares
```

In [21]:

```
# 03 - Crie uma FUNÇÃO que receba uma string como argumento e retorne a mesma string em
letras maiusculas .
# faça uma chamada a função, passando como paramentro uma string

def listaString (feira):
    print(feira.upper())
    return
listaString ('Laranja, banana, limão, uva, cenoura')
```

```
LARANJA, BANANA, LIMÃO, UVA, CENOURA
```

In [31]:

```
# 04 - Criar uma função que recebe como parâmetro uma lista de cinco elementos, acicion
ar outros dois elementos e imprimir:

def listaFeira (feira):
    print(feira.append('banana'))
    print(feira.append('cenoura'))

feira1= ['larana','limao','tomate','uva','maça']
listaFeira (feira1)
print(feira1)
```

```
None
None
['larana', 'limao', 'tomate', 'uva', 'maça', 'banana', 'cenoura']
```

In [36]:

```
# 5 - Criar uma função que receba um argumento formal e uma possível lista de elemento
s.
# Fazer duas chamadas a fução com apenas um elemento e na segunda com 4 elementos.
# Fazendo a chamada
def printNum(arg1, *lista):
    print (arg1)
    for i in lista:
        print(i)
    return;
printNum(150)
printNum('a','b','c','d')
```

```
150
a
b
c
d
```

In [40]:

```
# 6 - Crie uma função e atribua ser retoorno a uma variável chamada soma. a expressão v
ai receber dois números e retornar:

# expressões anonimas, tb lambda sem o uso da def

soma = lambda arg, arg1: arg + arg1
print ( 'A soma entre os argumentos é ', soma (556546, 7848993))
```

```
A soma entre os argumentos é  8405539
```

In [50]:

```
# 7 - Execute o código e certifique se que compreende a difernça entre a variavel golba
l e local

total  = 0

def soma( arg,  arg1):
    total = arg + arg1;
    print('Resultado dentro da Função é: ', total)
    return total

soma (155245, 6746836)
print ('Resultado fora da Função é:', total)
```

```
Resultado dentro da Função é:  6902081
Resultado fora da Função é: 0
```

In [51]:

```
# 8 - Crie uma função anônima para conversão de temperaturas em graus Celsius.
Celsius = [38.7, 35.6, 37.2, 38.8]
faherntheit = map(lambda x: (float(9)/5)*x + 32, Celsius)
print(list(faherntheit))
```

```
[101.66000000000001, 96.08, 98.96000000000001, 101.84]
```

In [52]:

```
# 9 - Criar um dicionário e listar os metodos e atributos deste dicionário:

dic = {'k1': 'São Paulo', 'k2':'Ceara'}
dir(dic)
```

Out[52]:

```
['__class__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__setitem__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'clear',
 'copy',
 'fromkeys',
 'get',
 'items',
 'keys',
 'pop',
 'popitem',
 'setdefault',
 'update',
 'values']
```

In [55]:

```
# 10 - import pandas as pd

import pandas as pd
pd.__version__
```

Out[55]:

```
'0.25.1'
```

In [56]:

```
# 11 - # Pandas é muito usado para analise de Dados, veja os métodos disponíveis em pan
das

import pandas as pd
dir(pd)
```

Out[56]:

```
['Categorical',
 'CategoricalDtype',
 'CategoricalIndex',
 'DataFrame',
 'DateOffset',
 'DatetimeIndex',
 'DatetimeTZDtype',
 'ExcelFile',
 'ExcelWriter',
 'Float64Index',
 'Grouper',
 'HDFStore',
 'Index',
 'IndexSlice',
 'Int16Dtype',
 'Int32Dtype',
 'Int64Dtype',
 'Int64Index',
 'Int8Dtype',
 'Interval',
 'IntervalDtype',
 'IntervalIndex',
 'MultiIndex',
 'NaT',
 'NamedAgg',
 'Period',
 'PeriodDtype',
 'PeriodIndex',
 'RangeIndex',
 'Series',
 'SparseArray',
 'SparseDataFrame',
 'SparseDtype',
 'SparseSeries',
 'Timedelta',
 'TimedeltaIndex',
 'Timestamp',
 'UInt16Dtype',
 'UInt32Dtype',
 'UInt64Dtype',
 'UInt64Index',
 'UInt8Dtype',
 '__builtins__',
 '__cached__',
 '__doc__',
 '__docformat__',
 '__file__',
 '__getattr__',
 '__git_version__',
 '__loader__',
 '__name__',
 '__package__',
 '__path__',
 '__spec__',
 '__version__',
 '_config',
 '_hashtable',
 '_lib',
 '_libs',
```

```
    '_np_version_under1p14',
    '_np_version_under1p15',
    '_np_version_under1p16',
    '_np_version_under1p17',
    '_tslib',
    '_typing',
    '_version',
    'api',
    'array',
    'arrays',
    'bdate_range',
    'compat',
    'concat',
    'core',
    'crosstab',
    'cut',
    'date_range',
    'datetime',
    'describe_option',
    'errors',
    'eval',
    'factorize',
    'get_dummies',
    'get_option',
    'infer_freq',
    'interval_range',
    'io',
    'isna',
    'isnull',
    'lreshape',
    'melt',
    'merge',
    'merge_asof',
    'merge_ordered',
    'notna',
    'notnull',
    'np',
    'offsets',
    'option_context',
    'options',
    'pandas',
    'period_range',
    'pivot',
    'pivot_table',
    'plotting',
    'qcut',
    'read_clipboard',
    'read_csv',
    'read_excel',
    'read_feather',
    'read_fwf',
    'read_gbq',
    'read_hdf',
    'read_html',
    'read_json',
    'read_msgpack',
    'read_parquet',
    'read_pickle',
    'read_sas',
    'read_spss',
    'read_sql',
```

```
 'read_sql_query',
 'read_sql_table',
 'read_stata',
 'read_table',
 'reset_option',
 'set_eng_float_format',
 'set_option',
 'show_versions',
 'test',
 'testing',
 'timedelta_range',
 'to_datetime',
 'to_msgpack',
 'to_numeric',
 'to_pickle',
 'to_timedelta',
 'tseries',
 'unique',
 'util',
 'value_counts',
 'wide_to_long']
```

In [64]:

```
# 12 - Crie uma função que recebe o arquivo com argumento e retorne um resumo estatisti
co descritivo
import pandas as pd
file_name = "dadosRH_modificado.csv"

def retornaArq(file_name):
return df.describe()

retornaArq(file_name)
```

```
  File "<ipython-input-64-085e122bb414>", line 6
    return df.describe()
         ^
IndentationError: expected an indented block
```

In [ ]: