In [2]:
```
a = 3
print(a)
a = 5
print(a)
```

```
3
5
```

In [3]:
```
print(a)
```

```
5
```

In [7]:
```
#calculate the number of seconds in 6 years
sec_per_min = 60
min_per_hour = 60
hour_per_day = 24
day_per_year = 365
total_secs = sec_per_min * min_per_hour * hour_per_day * day_per_year * 4
```

In [8]:
```
total_secs
```

Out[8]:
```
126144000
```

In [10]:
```
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [11]:
```
import urllib.request
urllib.request.urlretrieve("https://raw.githubusercontent.com/datasciencedojo/dat
titanic_df = pd.read_csv("./glo/titanic.csv")
```

In [12]:
```
titanic_df
```

Out[12]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	
...	...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	

891 rows × 12 columns

In [13]:
```
titanic_df.columns
```

Out[13]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [25]:
```
total_pasenger = titanic_df.PassengerId.count()
```

In [18]:
```
titanic_df.Survived.nunique()
```

Out[18]:
```
2
```

In [19]:
```
survive_df = titanic_df.Survived[titanic_df.Survived == 1]
```

In [24]:
```
survived_df = survive_df.count()
```

In [22]:
```
minor_df = titanic_df.Age[titanic_df.Age < 18]
```

In [23]:
```
minor_df.count()
```

Out[23]:
```
113
```

In [28]:
```
survived_fraction = survived_df  * 100 / total_pasenger
```

In [29]:
```
survived_fraction
```

Out[29]:
```
38.38383838383838
```

In [30]:
```
def salary(hourly_rate, number_hours, tax):
    #before tax
    pre_tax = hourly_rate * number_hours
    #after tax
    pro_tax = pre_tax * (1 - tax)
    return pro_tax
```

In [32]:
```
the_salary = salary(10, 200, 0.2)
```

In [33]:
```
the_salary
```

Out[33]:
```
1600.0
```

In [65]:
```
def house_cost(bed, bath, has_basement):
    value = 80000 + ((bed * 30000) + (bath * 10000))
    if has_basement:
        values = value + 40000
    else:
        values = value
    return values
```

In [67]:
```
house_cost(2, 1, True)
```

Out[67]: 190000

In [41]:
```
def get_cost(sqft_walls, sqft_ceiling, sqft_per_gallon, cost_per_gallon):
    wall_perimeter = sqft_walls + sqft_ceiling
    cost = wall_perimeter * cost_per_gallon / sqft_per_gallon
    return cost
```

In [43]:
```
cost_gallon = get_cost(432, 144, 400, 15)
```

In [44]:
```
import math
cost_gallon = math.ceil(cost_gallon)
```

In [45]:
```
cost_gallon
```

Out[45]: 22

In [50]:
```
round_it = abs(-3.14286)
```

In [51]:
```
round_it
```

Out[51]: 3.14286

In [52]:
```
print(3 * True)
print(-3.1 * True)
print(type("abc" * False))
print(len("abc" * False))
```

```
-3
```

In [59]:
```
print(len("abc" * False))
```

```
0
```

In [70]:
```
def cost_of_project(engraving, solid_gold):
    if solid_gold:
        cost = 100 + 10 * int(len(engraving))
    else:
        cost = 50 + 7 * int(len(engraving))
    return cost
```

In [76]:
```
cost_of_project("08/10/2000", True)
```

Out[76]: 200

In [78]:
```
def salary(salary):
    if salary < 12000:
        payment = salary * 0.25
    elif salary > 12000:
        payment = salary * 0.30
    return payment
```

In [80]:
```
salary(9000)
```

Out[80]:
```
2250.0
```

In [116]:
```
def get_grade(score):
    if 100 >= score >= 90:
        grade = "A"
    elif 90 >= score > 80:
        grade = "B"
    elif 80 >= score > 70:
        grade = "C"
    elif 70 >= score > 60:
        grade = "D"
    elif score < 60:
        grade = "F"
    return grade
```

In [120]:
```
get_grade(67)
```

Out[120]:
```
'D'
```

In [125]:
```
def get_water_bill(gallon_amount):
    if gallon_amount > 30000:
        water_bill = int(str(gallon_amount)[:2]) * 10
    elif 30000 >= gallon_amount >= 22000:
        water_bill = int(str(gallon_amount)[:2]) * 7
    elif 22000 >= gallon_amount >= 8000:
        water_bill = int(str(gallon_amount)[:2]) * 6
    elif 8000 >= gallon_amount >= 0:
        water_bill = int(str(gallon_amount)[:2]) * 5
    return water_bill
```

In [127]:
```
get_water_bill(25000)
```

Out[127]:
```
175
```

In [174]:
```
def get_phone_bill(gb):
    if gb <= 15:
        bill = 100
    elif gb > 15:
        substract = (gb - 15) * 1000
        now = (substract * 0.1)
        bill = (100 + now)
    return bill
```

In [176]:
```
the_bill = get_phone_bill(15.1)
```

In [177]:
```
round(the_bill)
```

Out[177]:
```
110
```

In [183]:
```
variable = "Lanrewaju"
print("my name is", "Adetunji Rilwan", variable)
```

```
my name is Adetunji Rilwan Lanrewaju
```

In [184]:
```
menu = ['stewed meat with onions', 'bean soup', 'risotto with trout and shrimp',
       'fish soup with cream and onion', 'gyro']
```

In [185]:
```
menu.remove("bean soup")
```

In [186]:
```
menu
```

Out[186]:
```
['stewed meat with onions',
 'risotto with trout and shrimp',
 'fish soup with cream and onion',
 'gyro']
```

In [205]:
```
num_customers = [137, 147, 135, 128, 170, 174, 165, 146, 126, 159,
                 141, 148, 132, 147, 168, 153, 170, 161, 148, 152,
                 141, 151, 131, 149, 164, 163, 143, 143, 166, 171]
```

In [189]:
```
avg_first_seven = sum(num_customers[0:7]) / 7
```

In [190]:
```
avg_first_seven
```

Out[190]:
```
150.85714285714286
```

In [206]:
```
avg_last_seven = sum(num_customers[-1:-2]) / 7
```

In [209]:
```
min(num_customers)
```

Out[209]:
```
126
```

In [210]:
```
flowers = "pink primrose,hard-leaved pocket orchid,canterbury bells,sweet pea,eng
```

In [211]:
```
flowers.split(",")
```

Out[211]:
```
['pink primrose',
 'hard-leaved pocket orchid',
 'canterbury bells',
 'sweet pea',
 'english marigold',
 'tiger lily',
 'moon orchid',
 'bird of paradise',
 'monkshood',
 'globe thistle']
```

In [227]:
```
alphabet = "A.B.C.D.E.F.G.H.I.J.K.L.M.N.O.P.Q.R.S.T.U.V.W.X.Y.Z"
address = "Mr. H. Potter,The cupboard under the Stairs,4 Privet Drive,Little Whir
```

In [233]:
```
address.split(",")
```

Out[233]:
```
['Mr. H. Potter',
 'The cupboard under the Stairs',
 '4 Privet Drive',
 'Little Whinging',
 'Surrey']
```

In [246]:
```
def test_rating(test_ratings):
    value = []
    for i in test_ratings:
        if i >= 4:
            value.append(i)
    return value
```

In [247]:
```
test_rating([3, 5, 7, 2])
```

Out[247]:
```
[5, 7]
```

In [244]:
```
test_rating([1, 2, 3, 4, 5])
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_2692\1705986837.py in <module>
----> 1 test_rating([1, 2, 3, 4, 5])

TypeError: 'list' object is not callable
```

In [288]:
```
the_list = [920344, 1043553, 1204334, 1458996, 1503323, 1593432, 1623463, 184306
the_list[(-1-7)]
```

Out[288]:
```
1204334
```

In [293]:
```
def function(the_list, years):
    growth = ((the_list[-1]) - (the_list[(-1 -years)])) / the_list[(-1 -years)]
    growths = (growth * 100)
    return growths
```

In [294]:
```
function(the_list, 7)
```

Out[294]:
```
66.15639847417742
```

In [35]:
```
nums = [7, 14, 37, 23, 89, 43, 77]
lucky = []
def has_lucky_number(nums):
    for num in nums:
        if num % 7 == 0:
            lucky.append(num)
    return lucky
```

In [36]:
```
has_lucky_number(nums)
```

Out[36]:
```
[7, 14, 77]
```

In [56]:
```
numes = [5, 1, 37, 23, 89, 43, 77]  #it will iterate all
def has_lucky_number(numes):
    """Return whether the given list of numbers is lucky. A lucky list contains
    at least one number divisible by 7.
    """
    for num in numes:
        if num % 7 == 0:
            return True
    return False
```

In [57]:
```
has_lucky_number(numes)
```

Out[57]:
```
True
```

In [ ]:
```
def has_lucky_number(nums):  # it will iterate once
    """Return whether the given list of numbers is lucky. A lucky list contains
    at least one number divisible by 7.
    """
    for num in nums:
        if num % 7 == 0:
            return True
        else:
            return False
```

In [26]:
```
value = []
def elementwise_greater_than(L, thresh):
    """Return a list with the same length as L, where the value at index i is
    True if L[i] is greater than thresh, and False otherwise.

    >>> elementwise_greater_than([1, 2, 3, 4], 2)
    [False, False, True, True]
    """
    for lis in L:
        if lis > thresh:
            return True
        else:
            return False
```

In [27]:
```
elementwise_greater_than([1, 2, 3, 4], 2)
```

Out[27]: False

In [49]:
```
def has_lucky_number(nums):
    """Return whether the given list of numbers is lucky. A lucky list contains
    at least one number divisible by 7.
    """
    return any([ num%7==0 for num in nums])
```

In [50]:
```
nums = [5, 14, 37, 23, 89, 43, 77]
```

In [51]:
```
has_lucky_number(nums)
```

Out[51]: True

In [61]:
```
meals = ["Amala", "Amala", "Bread", "Amala"]
def function(meals):
    for i in range(len(meals) - 1):
        if meals[i] == meals[i+1]:
            return True
    return False
```

In [62]:
```
function(meals)
```

Out[62]: True

In [13]:
```
claim = "Pluto is a planet!"
word = claim.endswith("planet!")
```

In [14]:
```
word
```

Out[14]: True

In [19]:
```
datestr = '1956-01-31'
year, month, day = datestr.split('-')
```

In [24]:
```
year, month, day
```

Out[24]:
```
('1956', '01', '31')
```

In [20]:
```
datestr
```

Out[20]:
```
'1956-01-31'
```

In [22]:
```
"/".join(datestr)
```

Out[22]:
```
'1/9/5/6/-/0/1/-/3/1'
```

In [23]:
```
"/".join([year, month, day])
```

Out[23]:
```
'1956/01/31'
```

In [34]:
```
a = "\n"
length = len(a)
```

In [35]:
```
length
```

Out[35]:
```
1
```

In [38]:
```
import math
round((math.pi), 3)
```

Out[38]:
```
3.142
```

In [39]:
```
print(math.pi, math.log(32, 2))
```

```
3.141592653589793 5.0
```

In [59]:
```
import numpy
rolls = numpy.random.randint([2, 3, 2])
rolls
```

Out[59]:
```
array([0, 1, 1])
```

In [62]:
```
numpy.array([14, 12, 14, 15, 13, 11, 14, 12, 15, 11]) + 10
```

Out[62]:
```
array([24, 22, 24, 25, 23, 21, 24, 22, 25, 21])
```

In [ ]:
```

```