```
user_name = "sai rishik"
```

```
user_name
```

```
     'sai rishik'
```

```
type(user_name)
```

```
     str
```

```
age = 21
```

```
type(age)
```

```
     int
```

```
tax = 23.55
```

```
type(tax)
```

```
     float
```

```
is_male = True
```

```
type(is_male)
```

```
     bool
```

```
#equality condition
5 == 4
```

```
     False
```

```
# not equal condition
5 != 4
```

```
     True
```

```
input("Please enter your age ")
```

```
     Please enter your age 33
     '33'
```

```
P= float(input("P: "))

R = 0.07
t = 5
A = P*(1+R)**t

print(A)
```

```
     P: 23.99
     33.64721601949301
```

```
P= float(input("P: "))

R = 0.07
t = 6
A = P*(1+R)**t

print(A)
```

```
     P: 100000
```

```
    150073.03518490004
```

```
P= float(input("P: "))
t= float(input("Year: "))

if P >= 100000:
  R = 0.07
else:
    R = 0.065

A = P*(1+R)**t


print(A)
```

```
    P: 100000
    Year: 6
    150073.03518490004
```

```
Budget = float(input("What is your budget:"))

if Budget >= 2000:
  print("Air travel")
elif Budget >= 1000 and Budget <= 1999:
  print("AC bus/ train travel")
elif Budget >= 500 and Budget <= 999:
  print("Non-AC bus/ train travel")
else:
  print("Please increase the budget")
```

```
    What is your budget:2000
    Air travel
```

```
Budget = float(input("What is your budget:"))
```

```
if Budget >= 2000:
  print("Air travel")
elif Budget >= 1000 and Budget <= 1999:
  print("AC bus/ train travel")
elif Budget >= 500 and Budget <= 999:
  print("Non-AC bus/ train travel")
else:
  print("Please increase the budget")
```

```
     What is your budget:400
     Please increase the budget
```

```
List_of_numbers = [12,34,56,78]
List_of_strings = ["Hello" , "Students" , "at" , "UBS"]
List_of_lists = [List_of_numbers , List_of_strings]
List_of_numbers.reverse()
```

```
type(List_of_numbers)
```

```
     list
```

```
List_of_numbers.reverse()
```

```
List_of_numbers = [12,34,56,78]
List_of_numbers.sort()
List_of_numbers.pop()
```

```
     78
```

```
List_of_numbers = [12,34,56,78]
List_of_numbers.reverse()
```

```
List_of_numbers

     [78, 56, 34, 12]

List_of_numbers[0]

     78


List_of_numbers[2]

     34


List_of_lists[1]

     ['Hello', 'Students', 'at', 'UBS']


List_of_lists[1][0]

     'Hello'


List_of_lists.reverse()


List_of_lists

     [['Hello', 'Students', 'at', 'UBS'], [78, 56, 34, 12]]


List_of_lists[1][0] = "Business Analytics"


List_of_lists

     [['Hello', 'Students', 'at', 'UBS'], ['Business Analytics', 56, 34, 12]]


tuple_of_numbers = (12,33,1213,233)
```

```
tuple_of_numbers[3]
```

```
    233
```

```
# tuple is an immutable object
```

```
tuple_of_numbers[1:3]
```

```
    (33, 1213)
```

```
my_dict = {
    "Sai" : "male"


}
```

```
print(my_dict)
```

```
    {'Sai': 'male'}
```

```
calendar = {"JAN":["New Years" , "Makarshankranti" , "Republic Day"] , "FEB": "No holidays" , "MAR":["Mahashivratri" , "Holi"] , "APR
```

```
print(calendar)
```

```
    {'JAN': ['New Years', 'Makarshankranti', 'Republic Day'], 'FEB': 'No holidays', 'MAR': ['Mahashivratri', 'Holi'], 'APR': ['good
```

```
calendar
```

```
{'APR': ['good friday , ugadi , mahavir jayanti'],
 'FEB': 'No holidays',
 'JAN': ['New Years', 'Makarshankranti', 'Republic Day'],
 'MAR': ['Mahashivratri', 'Holi']}
```

```
print(calendar)
```

```
{'JAN': ['New Years', 'Makarshankranti', 'Republic Day'], 'FEB': 'No holidays', 'MAR': ['Mahashivratri', 'Holi'], 'APR': ['good
```

```
calendar.keys()
```

```
dict_keys(['JAN', 'FEB', 'MAR', 'APR'])
```

```
enter month: JAN
```

```
calendar.keys()
```

```
dict_keys(['JAN', 'FEB', 'MAR', 'APR'])
```

```
calendar["JAN"]
```

```
['New Years', 'Makarshankranti', 'Republic Day']
```

```
Month = calendar["input("")"]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-53-da45248a521f> in <module>()
----> 1 Month = calendar["input("")"]
```

```
Month = calendar["JAN"]
```

```
Month
```

```
['New Years', 'Makarshankranti', 'Republic Day']
```

```
Month = calendar[input("")]
print(Month)
```

```
JAN
['New Years', 'Makarshankranti', 'Republic Day']
```

```
Month = input("please enter the month to view the holidays: ")
print(calendar[Month])
```

```
please enter the month to view the holidays: jan
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-64-8dfb955d44b7> in <module>()
      1 Month = input("please enter the month to view the holidays: ")
----> 2 print(calendar[Month])

KeyError: 'jan'
```

SEARCH STACK OVERFLOW

```
Month.lower()
```

```
     'jan'
```

```
''' Savio says, "Be best in python programming"'''
```

```
     ' Savio says, "Be best in python programming"'
```

```
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import KMeans
```

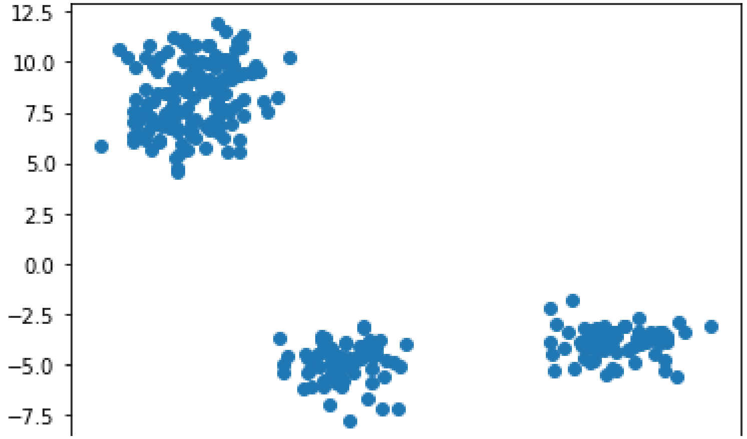
```
x,y = make_blobs(n_samples = 300, centers = 4)
```

```
x
```

```
     array([[-6.63599218,  6.4031751 ],
            [-6.91142558,  5.96980078],
            [-7.78590723,  8.08596596],
            [-2.83977268, -4.5518044 ],
            [-8.20814426,  6.43146459],
            [-6.41772794,  9.31011686],
            [-0.84071889, -5.6038536 ],
            [-5.88338201,  8.62522733],
            [-2.14576369, -5.16286689],
            [ 5.82835701, -5.46233981],
            [-6.1249959 ,  9.1768255 ],
            [ 6.53159779, -4.28583499],
            [-7.52263968,  6.90193296],
            [-1.17432342, -4.57155722],
            [-1.44205969, -4.43147704],
            [-2.42486097, -5.44665987],
            [-7.83024073,  7.24082153],
            [-5.7411618 ,  9.15681721],
            [-5.29051129,  7.82846317],
            [-7.90782859, 10.83584216],
            [-6.53704446, 10.79754528],
            [ 4.90284485, -5.1577115 ],
            [-6.11057532,  8.70477844],
            [-7.30859209,  7.03413737],
            [-3.04910311, -5.25903417],
            [ 4.15663439, -3.90282186],
            [ 5.11207521, -4.10146562],
            [-6.22463831,  5.75780162],
            [ 5.5814312 , -3.16582485],
            [ 6.55970538, -4.18027654],
            [-7.82651272,  6.81338526],
            [ 4.72751134, -3.39884209],
            [-6.64413683,  7.02053762],
            [-7.76756281,  9.88980999],
            [-1.41980381, -3.15676329],
            [ 6.11638149, -3.55905242],
            [ 6.12848805, -5.30881013],
            [-7.13647354,  7.68860618],
            [-2.27919262, -4.92207811],
            [-0.97599393, -3.83912458],
            [-5.91835075,  6.79216339],
            [-1.62442299, -4.64456565],
            [ 5.5917015 , -4.24665414],
            [-6.08210657,  6.65749623],
            [ 4.26917063, -5.30766043],
            [-5.10897865, 10.72010662],
            [-5.60651146, 10.11641132],
            [-6.89423323, 10.02286904],
            [-7.68764588,  6.92001363],
            [-6.80998098,  9.01577746],
            [-3.83655689, -5.37162632],
            [-2.67388919, -6.07411509],
            [-7.07299588,  6.51534296],
            [-6.69788737,  6.61504231],
            [ 6.11461087, -3.40268218],
            [-6.54655545,  9.2090644 ],
            [-5.24002791, 11.01043437],
            [ 5.25041359, -3.73603883]
```

```
plt.scatter(x[:,0],x[:,1])
```

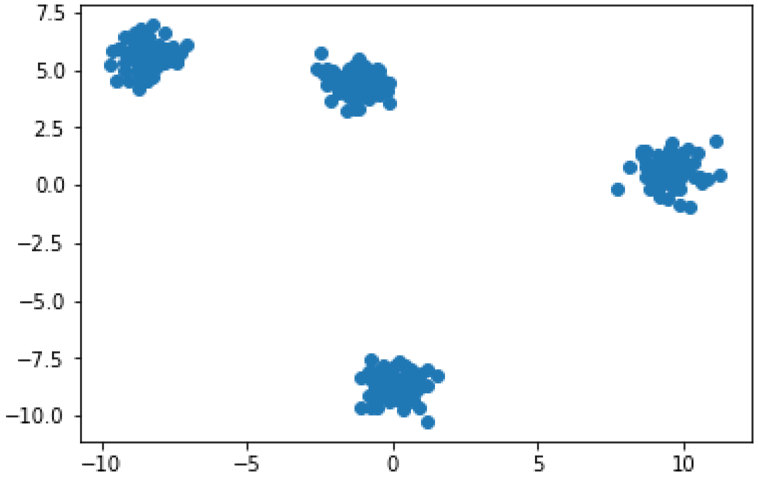
```
    <matplotlib.collections.PathCollection at 0x7f1737919b10>
```



```
coordinates,clusters = make_blobs(n_samples = 300, centers = 4, cluster_std=0.6, random_state= 7)
```

```
plt.scatter(coordinates[:,0],coordinates[:,1])
```

```
    <matplotlib.collections.PathCollection at 0x7f17371e2690>
```



```
clusters
```

```
    array([2, 3, 2, 1, 2, 0, 0, 3, 2, 0, 2, 0, 0, 1, 1, 1, 1, 0, 3, 3, 2, 3,
           2, 2, 2, 3, 3, 1, 0, 1, 3, 3, 3, 0, 0, 0, 1, 2, 3, 1, 3, 3, 3, 1,
       3, 2, 2, 3, 3, 2, 3, 0, 0, 3, 3, 3, 1, 2, 1, 2, 1, 1, 2, 1, 1, 3,
       2, 0, 2, 3, 0, 1, 1, 3, 2, 1, 2, 2, 1, 1, 2, 3, 0, 3, 3, 3, 1, 3,
       0, 0, 1, 3, 1, 0, 3, 3, 1, 0, 0, 0, 2, 1, 2, 1, 1, 1, 2, 1, 2, 1,
       2, 2, 1, 2, 0, 3, 1, 2, 2, 0, 0, 3, 1, 3, 1, 0, 1, 2, 2, 3, 2, 0,
       3, 0, 3, 3, 3, 1, 3, 2, 2, 0, 0, 3, 2, 0, 2, 3, 2, 0, 0, 0, 2, 2,
       2, 2, 0, 3, 0, 2, 3, 0, 3, 1, 2, 2, 1, 0, 0, 2, 2, 2, 3, 3, 1, 1,
       1, 3, 0, 1, 2, 2, 0, 1, 2, 2, 2, 2, 2, 0, 2, 1, 0, 1, 0, 1, 1, 3,
       3, 0, 3, 1, 3, 1, 0, 3, 0, 0, 0, 0, 1, 1, 3, 1, 1, 3, 1, 3, 0, 0,
       0, 3, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 2, 0, 3, 1, 0, 2, 1, 1, 2, 1,
       0, 3, 3, 3, 0, 0, 3, 3, 2, 2, 1, 1, 0, 3, 1, 3, 2, 2, 0, 2, 2, 2,
       3, 0, 1, 1, 3, 1, 2, 0, 2, 1, 0, 1, 2, 0, 0, 3, 0, 1, 3, 0, 3, 3,
       2, 1, 3, 2, 1, 2, 2, 3, 3, 1, 2, 1, 0, 0])
```

```
kmeans_clusters = Kmeans(n_clusters = 4)
kmeans_clusters.fit(coordinates)
y_kmeans = kmeans_clusters.predict(coordinates)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-be0f8f47371c> in <module>()
----> 1 kmeans_clusters = KMeans(n_clusters = 4)
      2 kmeans_clusters.fit(coordinates)
      3 y_kmeans = kmeans_clusters.predict(coordinates)

NameError: name 'KMeans' is not defined
```

SEARCH STACK OVERFLOW