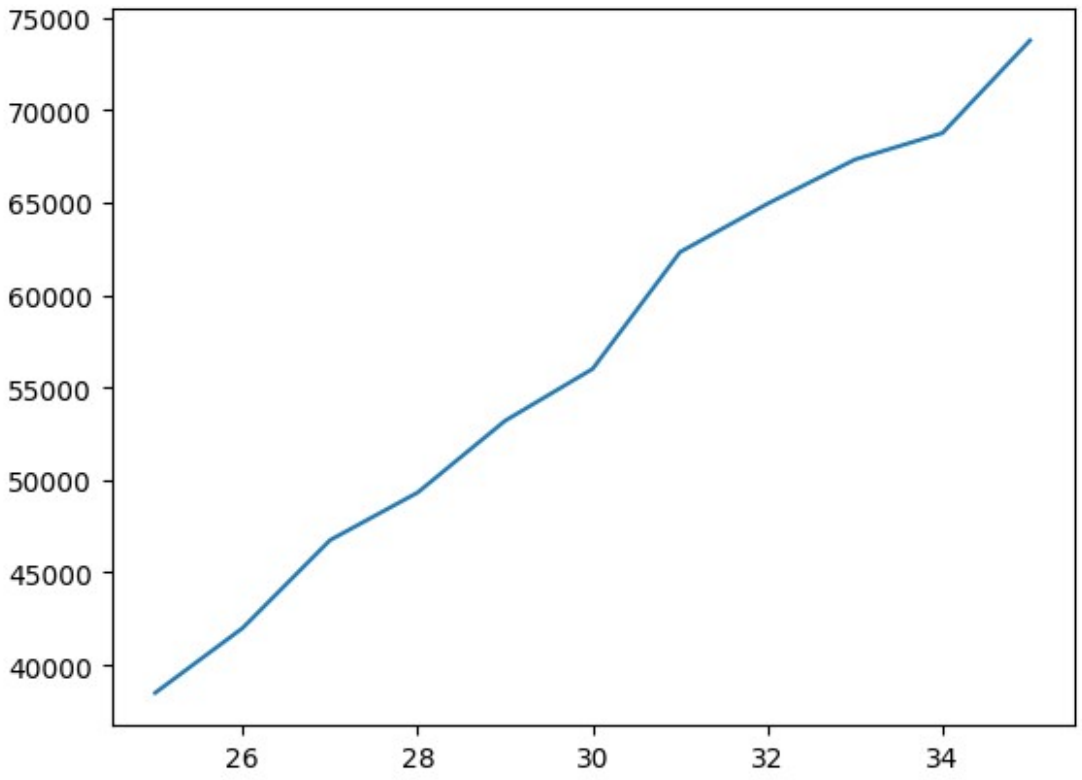
```
from matplotlib import pyplot as plt
import numpy as np
import pandas as pd
from collections import Counter
```

Plotting Data

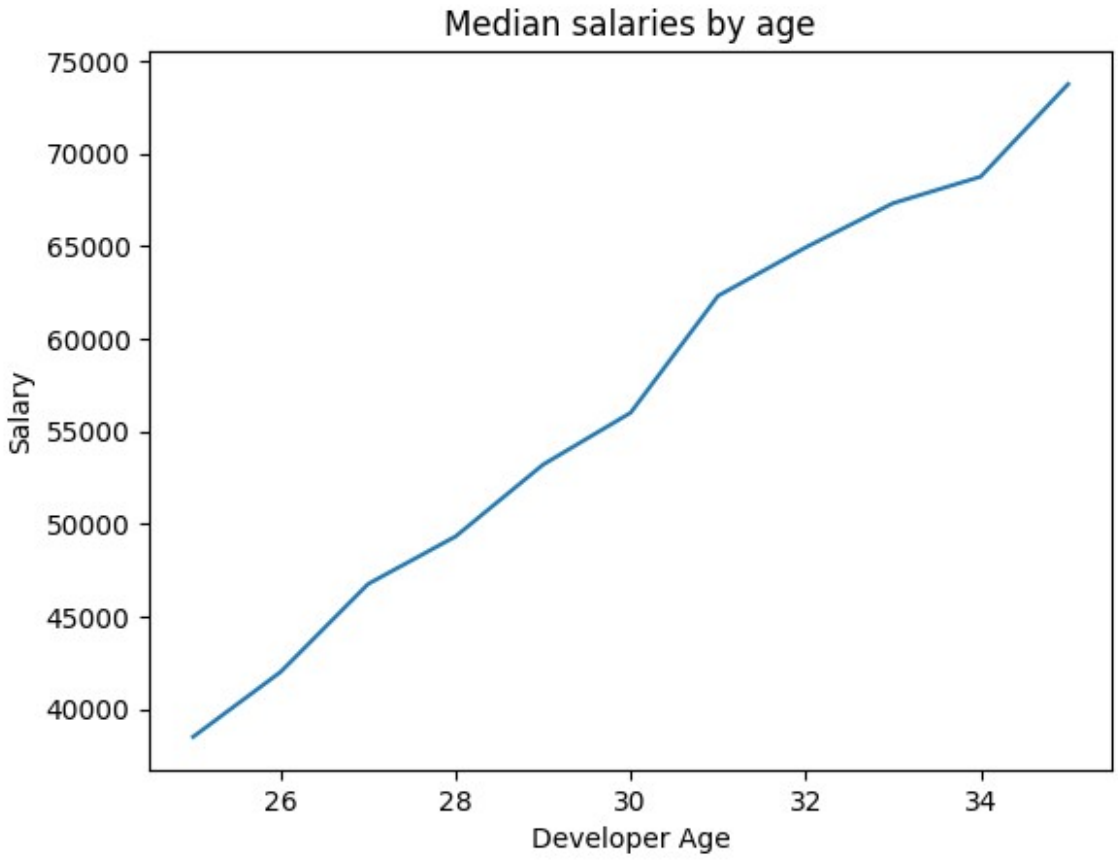
```
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]

plt.plot(dev_age, dev_salary)


plt.show()
```



giving Tittle and Label to the plot

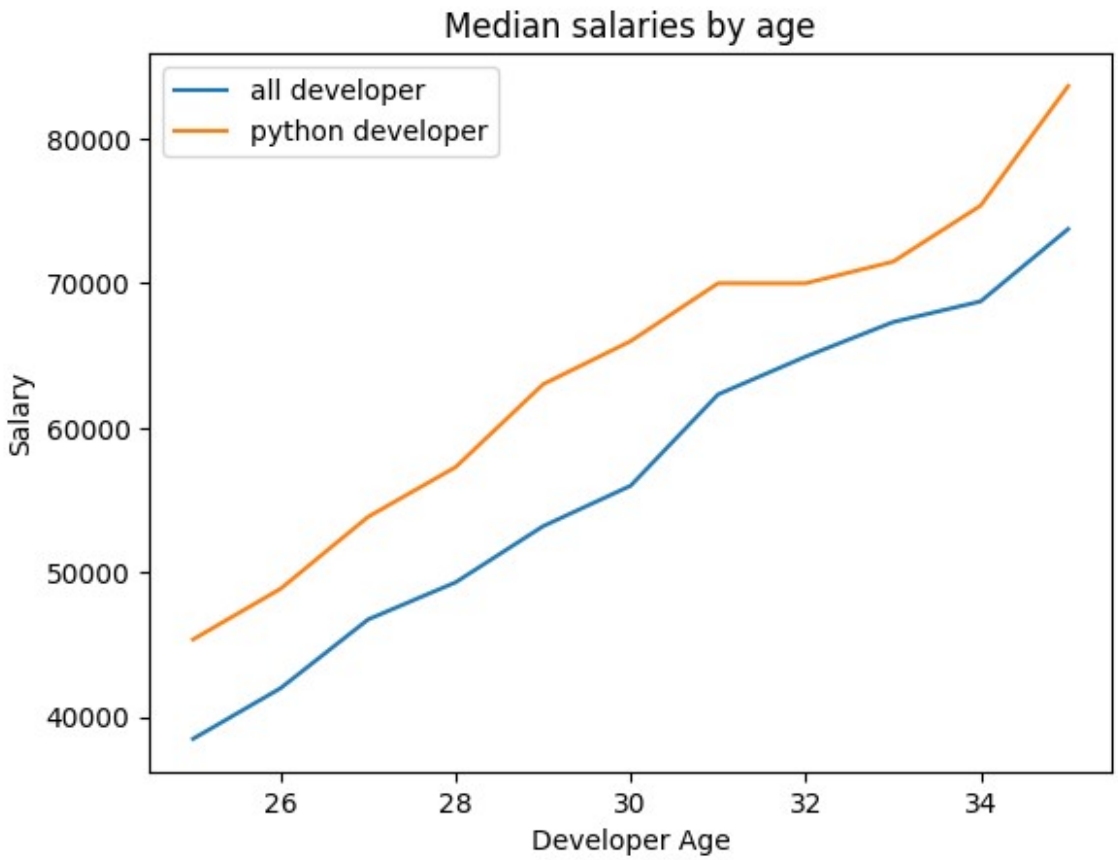
```
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]

plt.plot(dev_age, dev_salary)
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
```

```
plt.ylabel("Salary")
plt.show()
```



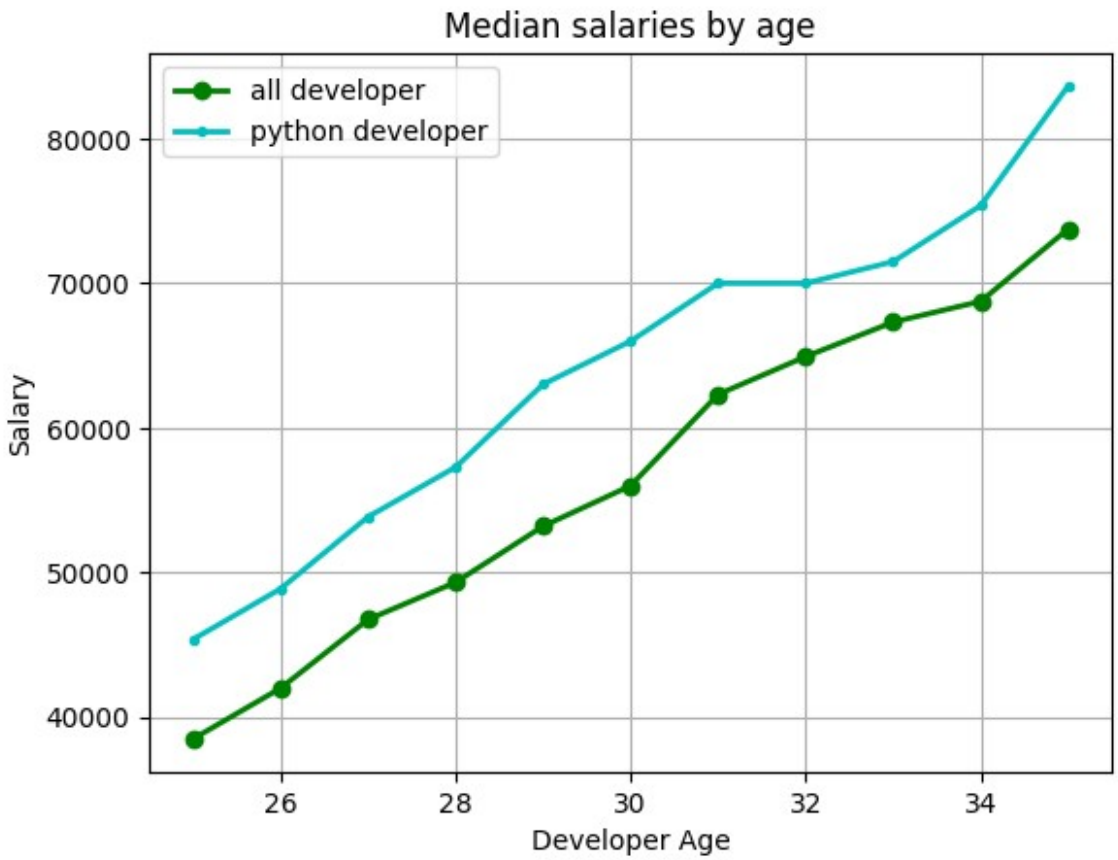
using of legend for multiple plot line

```
# Median Python Developer Salaries by Age
py_dev_salary = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]

plt.plot(dev_age, dev_salary,label = "all developer")
plt.plot(dev_age,py_dev_salary,label = "python developer")
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
plt.ylabel("Salary")
plt.legend() # for labeling every individual line
plt.show()
```



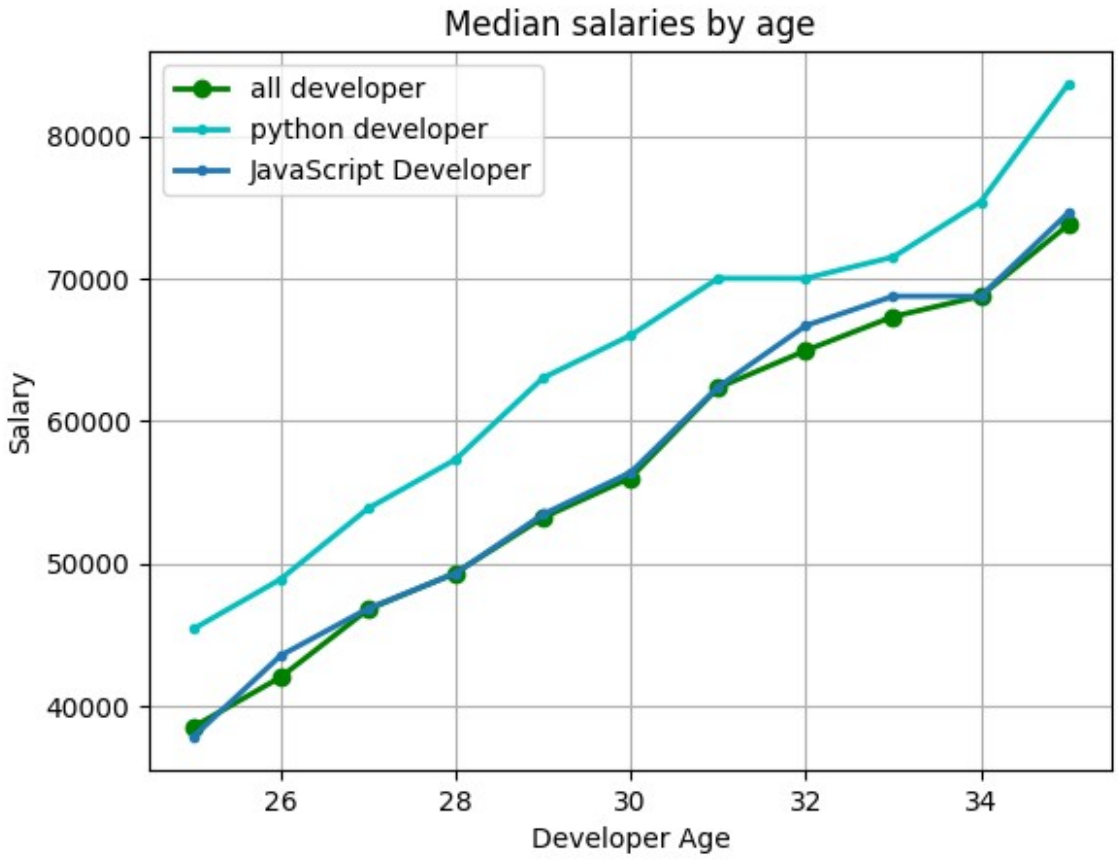
Formatting Plot

```
py_dev_salary = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]

plt.plot(dev_age, dev_salary,color = "g",marker = "o",lw = "2",label =
"all developer")
plt.plot(dev_age,py_dev_salary,label = "python developer",color =
"c",marker = ".",lw = "2")
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
plt.ylabel("Salary")
#for adding a grid in
plt.grid(True)
plt.legend() # for labeling every individual line
plt.show()
```



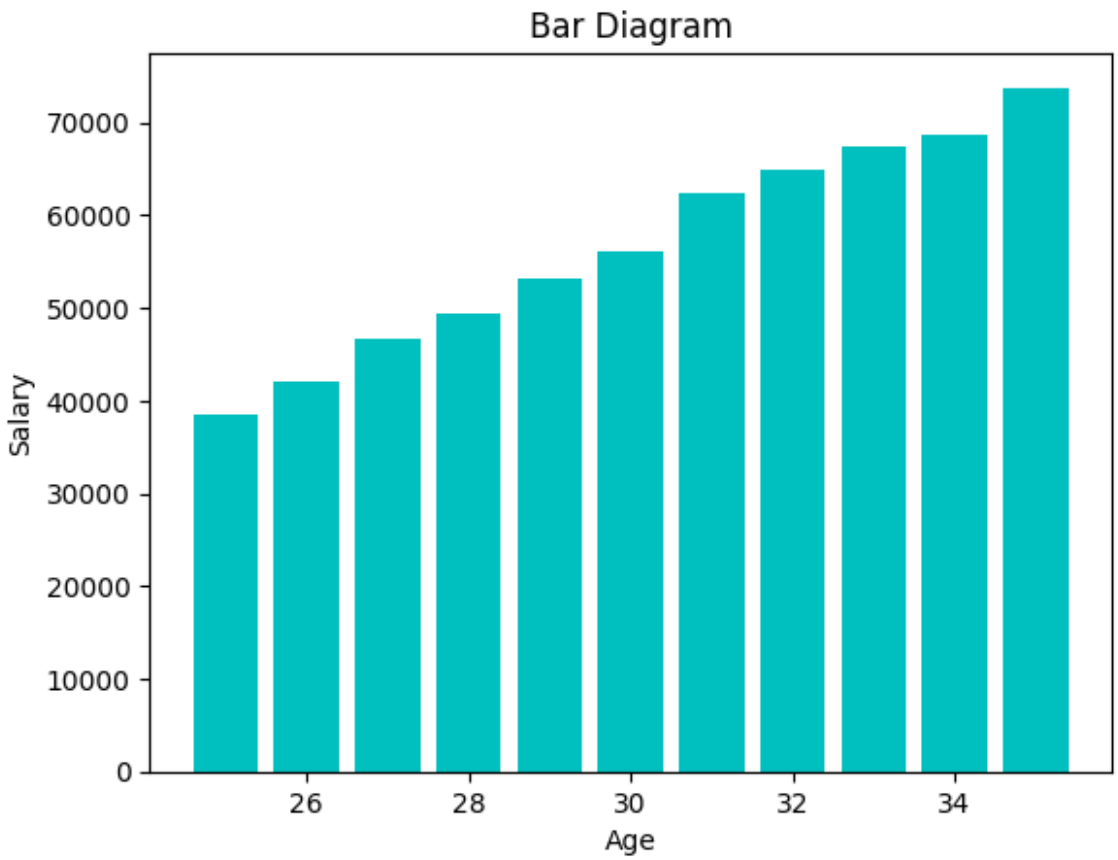

```
py_dev_salary = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]
# Median JavaScript Developer Salaries by Age
js_dev_salary = [37810, 43515, 46823, 49293, 53437,
            56373, 62375, 66674, 68745, 68746, 74583]



plt.plot(dev_age, dev_salary,color = "g",marker = "o",lw = "2",label =
"all developer")
plt.plot(dev_age,py_dev_salary,label = "python developer",color =
"c",marker = ".",lw = "2")
plt.plot(dev_age,js_dev_salary,label = "JavaScript Developer",marker =
".",lw = "2")
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
plt.ylabel("Salary")
#for adding a grid in
plt.grid(True)
```

```
plt.legend() # for labeling every individual line
plt.show()
```



Bar Charts

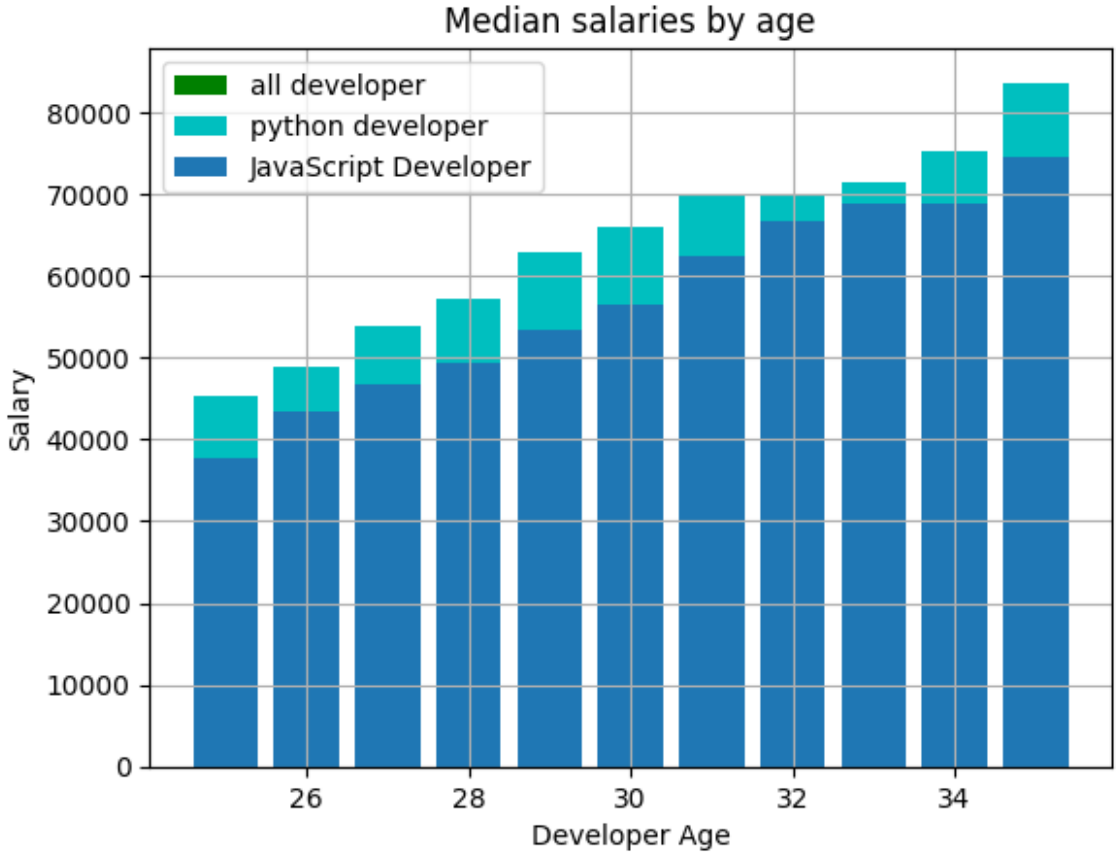
```
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]

plt.bar(dev_age, dev_salary,color = "c")
plt.xlabel ("Age")
plt.ylabel ("Salary")
plt.title("Bar Diagram")
plt.show()
```



Multiple bar chart

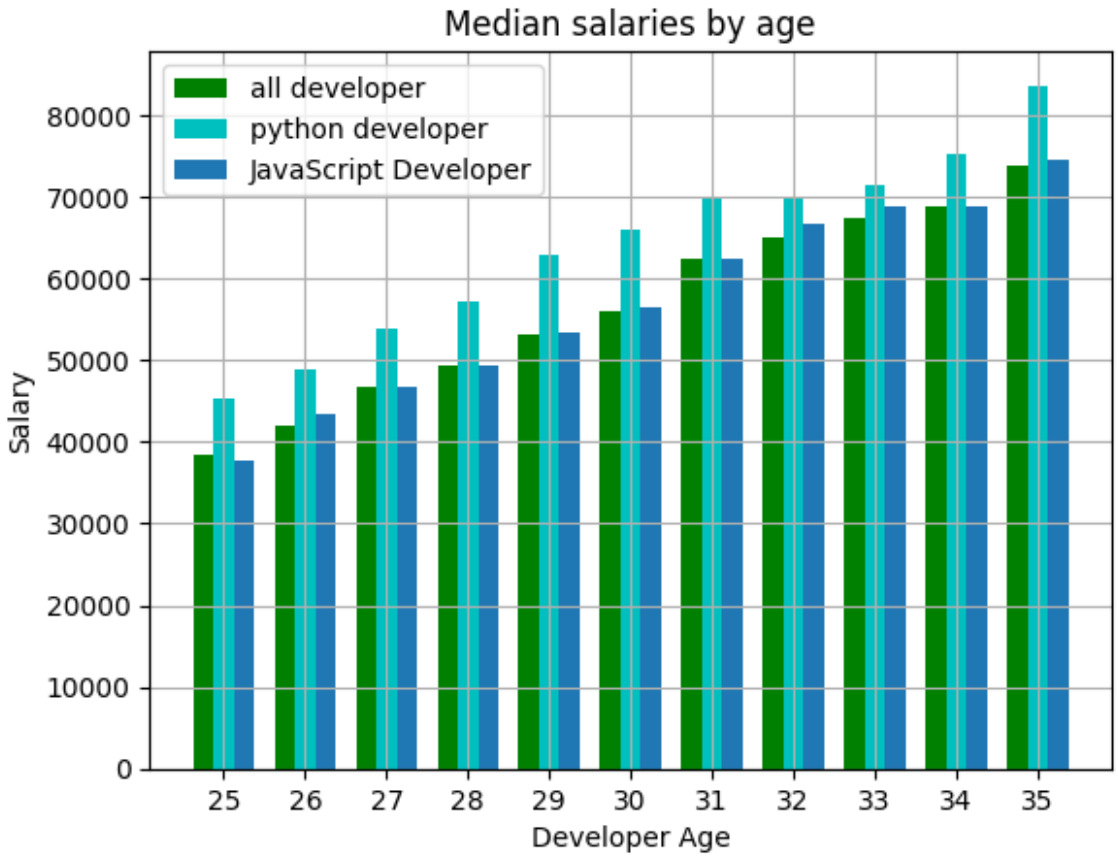
```
py_dev_salary = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]
# Median JavaScript Developer Salaries by Age
js_dev_salary = [37810, 43515, 46823, 49293, 53437,
            56373, 62375, 66674, 68745, 68746, 74583]



plt.bar(dev_age, dev_salary,color = "g",label = "all developer")
plt.bar(dev_age,py_dev_salary,label = "python developer",color = "c")
plt.bar(dev_age,js_dev_salary,label = "JavaScript Developer")
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
plt.ylabel("Salary")
#for adding a grid in
plt.grid(True)
plt.legend() # for labeling every individual line
plt.show()
```

```
# by this bar we can not see javascript Developer. They overlapped by
another.
```




```
#So now we are trying to do separate bar chart here
py_dev_salary = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
dev_age = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_salary = [38496, 42000, 46752, 49320, 53200,
              56000, 62316, 64928, 67317, 68748, 73752]
# Median JavaScript Developer Salaries by Age
js_dev_salary = [37810, 43515, 46823, 49293, 53437,
            56373, 62375, 66674, 68745, 68746, 74583]
wid = 0.25
index = np.arange(len(dev_age))

plt.bar(index - wid, dev_salary,width = wid,color = "g",label = "all
developer")
plt.bar(index,py_dev_salary,width = wid,label = "python
developer",color = "c")
plt.bar(index + wid,js_dev_salary,width = wid,label = "JavaScript
Developer")
```

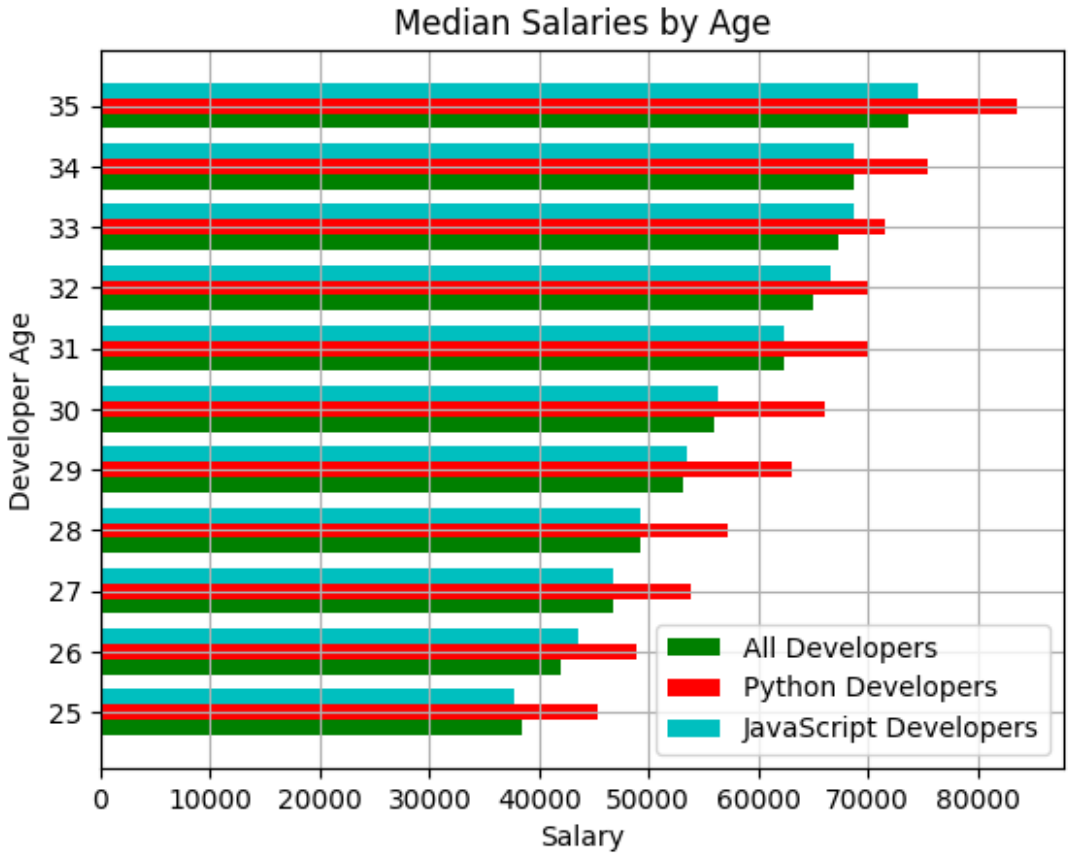
```
plt.title("Median salaries by age")
plt.xlabel("Developer Age")
plt.ylabel("Salary")
#plt.xticks(ticks = index,labels
=["a","b","c","d","e","f","g","h","i","j","k"] )
plt.xticks(ticks = index,labels =dev_age)
#xticks is very flexible. we can set labels by our wish and ticks
against another value which ticked that
#for adding a grid in
plt.grid(True)
plt.legend() # for labeling every individual line
plt.show()
```



Horizontal Bar chat

```
plt.barh(index - wid, dev_salary, height=wid, color="g", label="All
Developers")
plt.barh(index, py_dev_salary, height=wid, color="r", label="Python
Developers")
plt.barh(index + wid, js_dev_salary, height=wid, color="c",
label="JavaScript Developers")
```

```
plt.title("Median Salaries by Age")
plt.xlabel("Salary")
plt.ylabel("Developer Age")
plt.yticks(ticks=index, labels=dev_age)
plt.grid(True)
plt.legend()

plt.show()
```




```
df = pd.read_csv("G:\Self\Python\Matplotlib\dataset\data.csv")
df
```

	Responder_id	LanguagesWorkedWith
0	1	HTML/CSS;Java;JavaScript;Python
1	2	C++;HTML/CSS;Python
2	3	HTML/CSS
3	4	C;C++;C#;Python;SQL
4	5	C++;HTML/CSS;Java;JavaScript;Python;SQL;VBA
...	...	...
87564	88182	HTML/CSS;Java;JavaScript
87565	88212	HTML/CSS;JavaScript;Python
87566	88282	Bash/Shell/PowerShell;Go;HTML/CSS;JavaScript;W...

```
87567         88377                       HTML/CSS;JavaScript;Other(s):
87568         88863  Bash/Shell/PowerShell;HTML/CSS;Java;JavaScript...

[87569 rows x 2 columns]
```

```
df["LanguagesWorkedWith"]
```

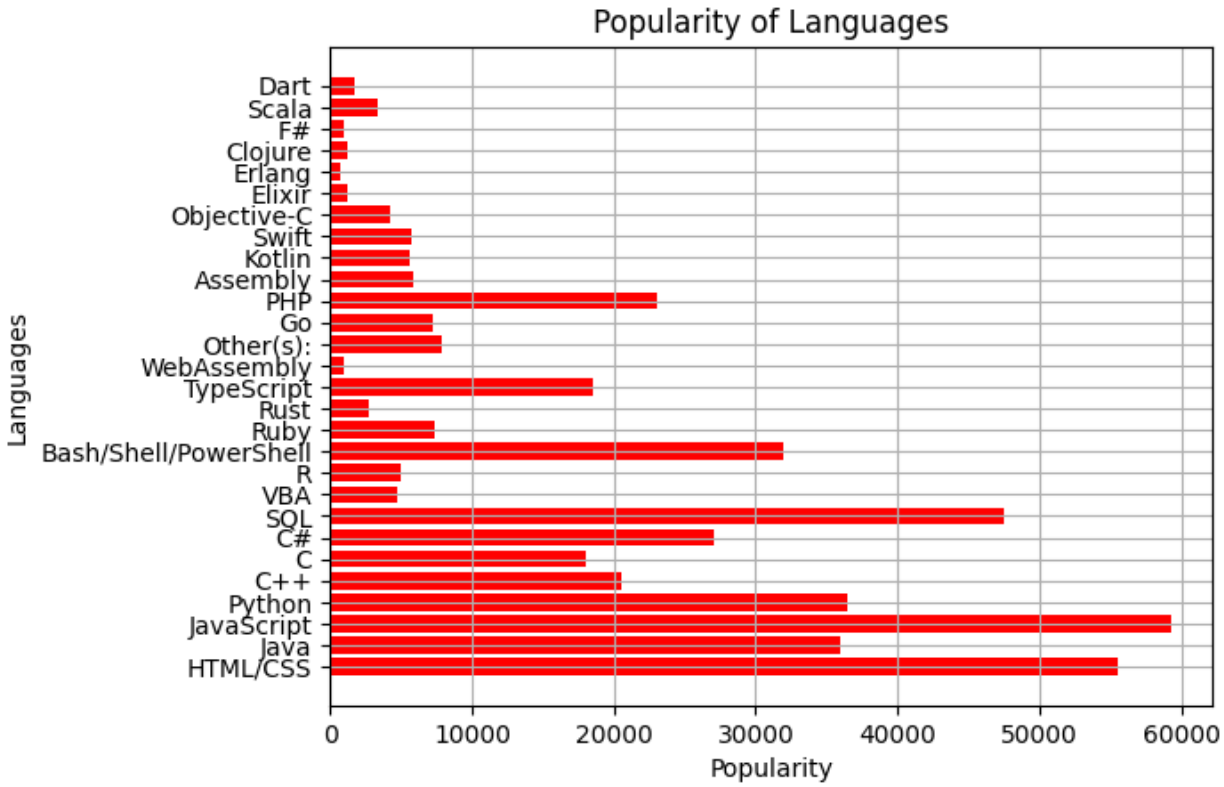
```
0                           HTML/CSS;Java;JavaScript;Python
1                                     C++;HTML/CSS;Python
2                                                HTML/CSS
3                                     C;C++;C#;Python;SQL
4              C++;HTML/CSS;Java;JavaScript;Python;SQL;VBA
                                  ...
87564                            HTML/CSS;Java;JavaScript
87565                          HTML/CSS;JavaScript;Python
87566    Bash/Shell/PowerShell;Go;HTML/CSS;JavaScript;W...
87567                        HTML/CSS;JavaScript;Other(s):
87568    Bash/Shell/PowerShell;HTML/CSS;Java;JavaScript...
Name: LanguagesWorkedWith, Length: 87569, dtype: object
```

```
lang_response = df["LanguagesWorkedWith"]
lang_counter = Counter()
for response in lang_response:
    lang_counter.update(response.split(";"))
language = []
popularity = []
for item in lang_counter:
    language.append(item)
    popularity.append(lang_counter[item])
dfn = {"language":language,"popularity": popularity}
DF =pd.DataFrame(dfn)
DF
```

	language	popularity
0	HTML/CSS	55466
1	Java	35917
2	JavaScript	59219
3	Python	36443
4	C++	20524
5	C	18017
6	C#	27097
7	SQL	47544
8	VBA	4781
9	R	5048
10	Bash/Shell/PowerShell	31991
11	Ruby	7331
12	Rust	2794
13	TypeScript	18523
14	WebAssembly	1015
15	Other(s):	7920

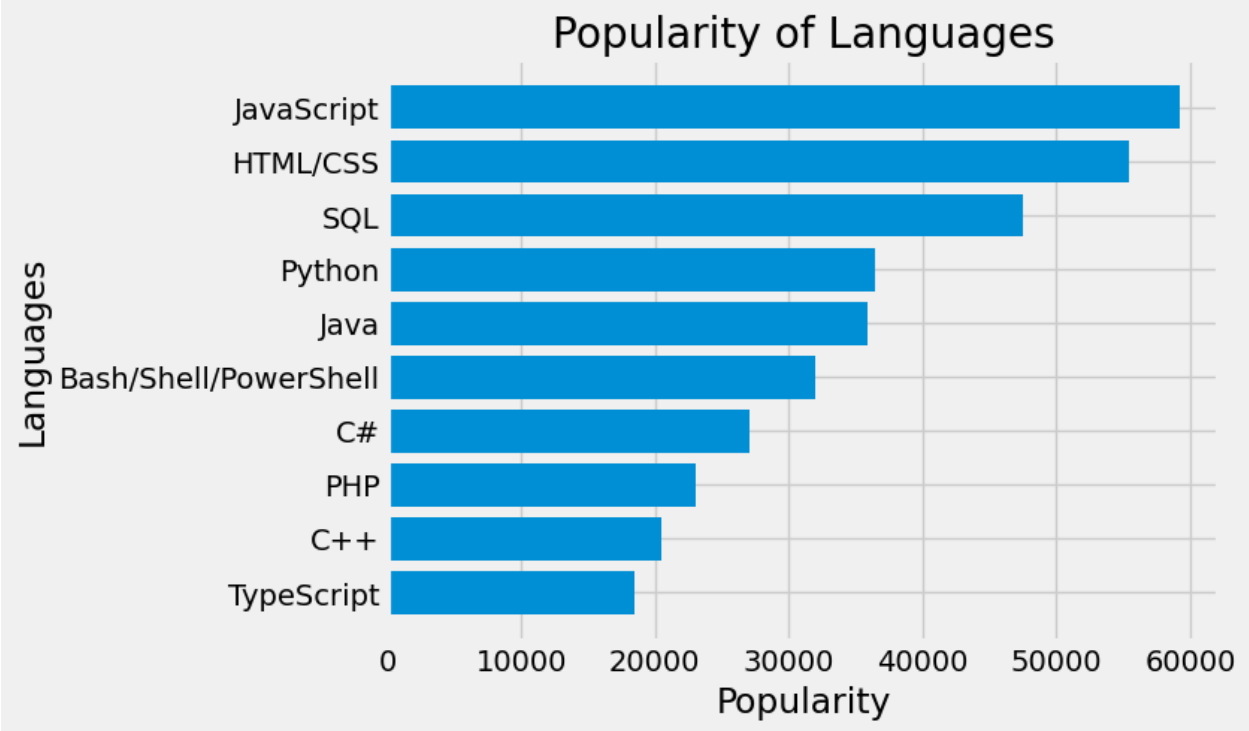
```
16                      Go         7201
17                     PHP        23030
18                Assembly         5833
19                  Kotlin         5620
20                   Swift         5744
21             Objective-C         4191
22                  Elixir         1260
23                  Erlang          777
24                 Clojure         1254
25                      F#          973
26                   Scala         3309
27                    Dart         1683
```

```
lang = DF["language"]
user = DF["popularity"]

plt.barh(lang,user,color = "r")
plt.title("Popularity of Languages")
plt.xlabel ("Popularity")
plt.ylabel ("Languages")
plt.grid()
plt.show()
```



finding most common languages from the file

```
plt.style.use("fivethirtyeight")
lang_response = df["LanguagesWorkedWith"]
lang_counter = Counter()
for response in lang_response:
    lang_counter.update(response.split(";"))
language = []
popularity = []
for item in lang_counter.most_common(10):
    language.append(item[0])
    popularity.append(item[1])
language.reverse()
popularity.reverse()
plt.barh(language,popularity)
plt.title("Popularity of Languages")
plt.xlabel ("Popularity")
plt.ylabel ("Languages")
plt.show()
```

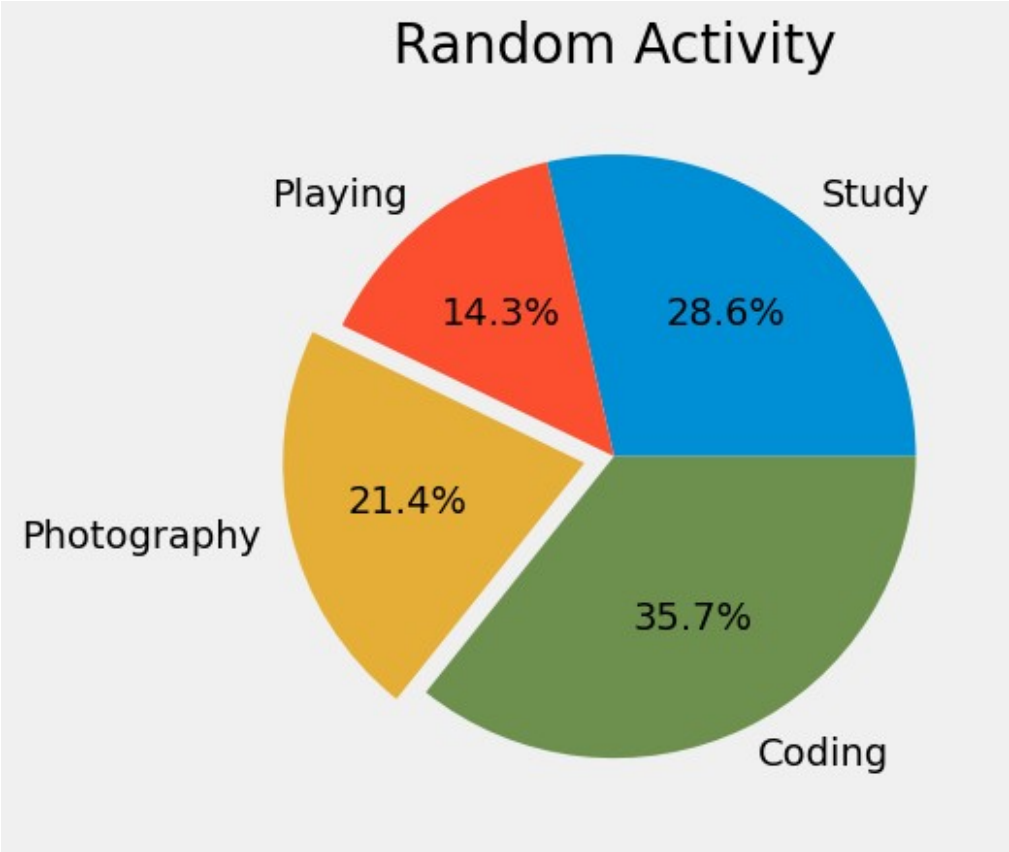


Pie Chart

Don't use piechart if you have more than 5 data. It seems massy. Use Bar char instead of Pie chart

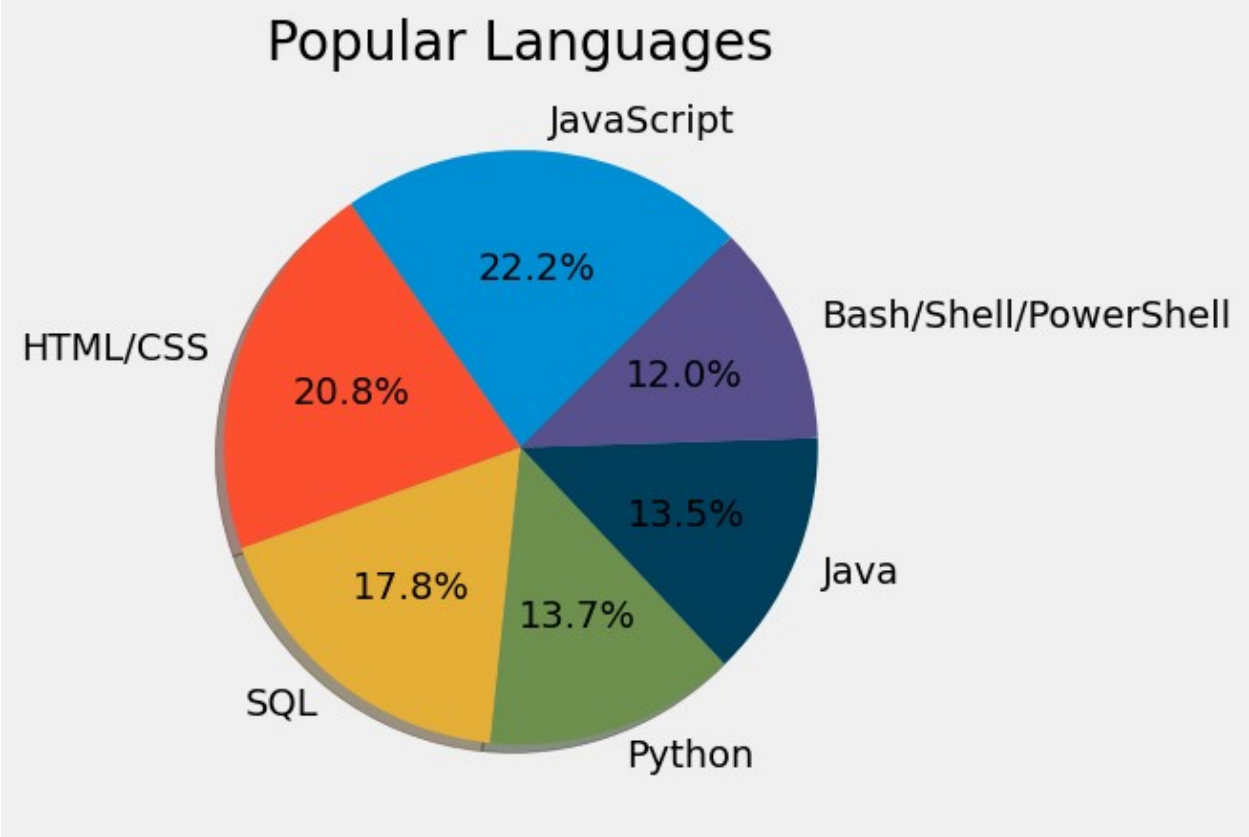
```
# Colors:
# Blue = #008fd5
```

```
# Red = #fc4f30
# Yellow = #e5ae37
# Green = #6d904f
#midnight green = #003F5C
#Purple Navy = #58508D

activity = [40,20,30,50]
labels = ["Study","Playing","Photography","Coding"]
colors = ["#008fd5","#fc4f30","#e5ae37","#6d904f","#003F5C","#58508D"]
plt.pie(activity,labels = labels,colors = colors,explode =
(0,0,.1,0),autopct="%1.1f%%")
plt.title("Random Activity")
plt.show()
```




```
language_P = []
user_P = []
for item in lang_counter.most_common(6):
    language_P.append(item[0])
    user_P.append(item[1])
language_P
user_P
plt.pie(user_P,labels = language_P,colors =colors,autopct="%1.1f%
%",shadow = True,startangle = 45 )
```

```
plt.title("Popular Languages")
plt.show()
```



Stack Plot

when you want to show the total and individual contributions of multiple components over a continuous interval, such as time.

Dataset

minutes = [1, 2, 3, 4, 5, 6, 7, 8, 9]

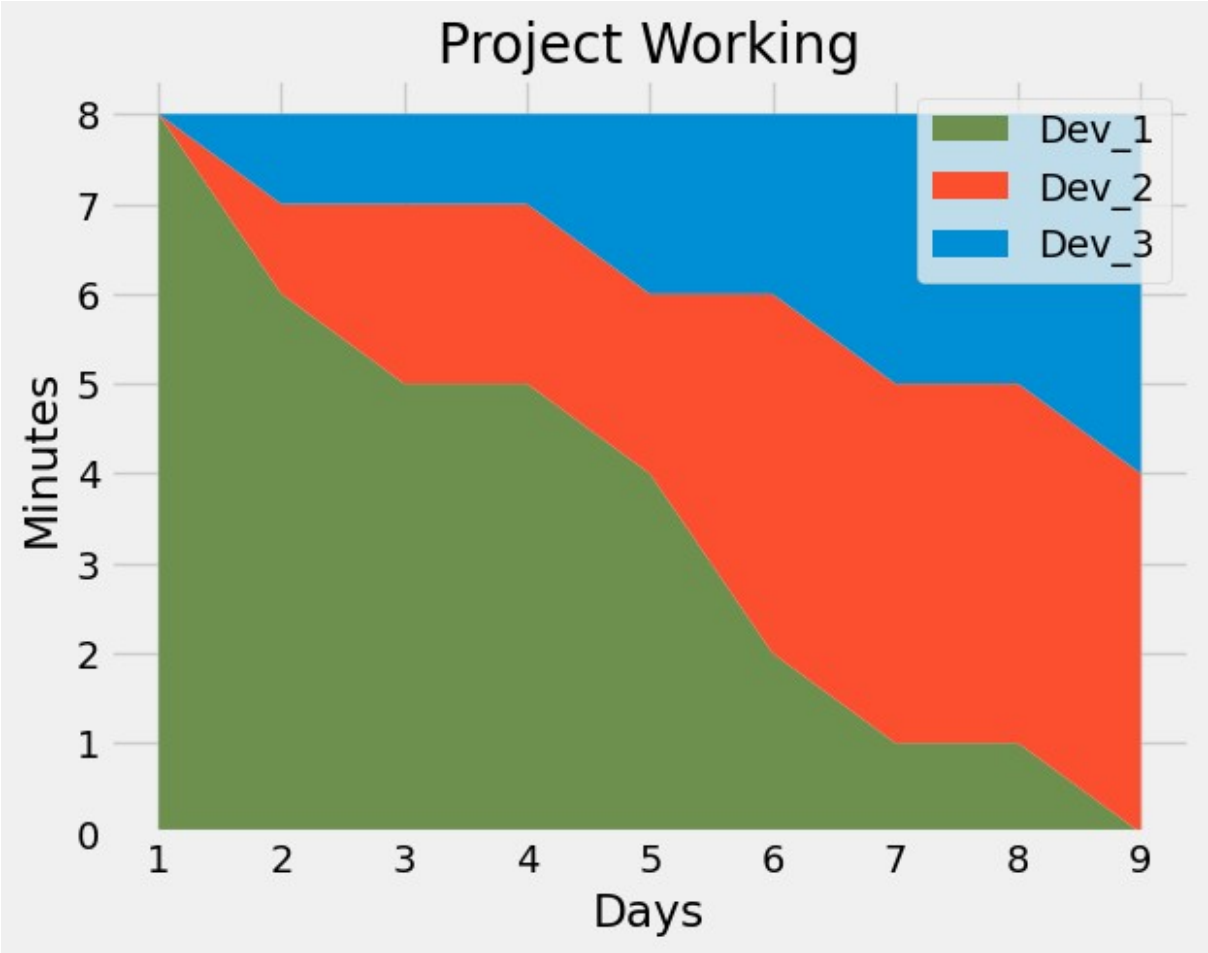
Dev_1 = [8, 6, 5, 5, 4, 2, 1, 1, 0]

Dev_2 = [0, 1, 2, 2, 2, 4, 4, 4, 4]

Dev_3 = [0, 1, 1, 1, 2, 2, 3, 3, 4]

```
days = [1, 2, 3, 4, 5, 6, 7, 8, 9]

Dev_1 = [8, 6, 5, 5, 4, 2, 1, 1, 0] # unit: minutes
Dev_2 = [0, 1, 2, 2, 2, 4, 4, 4, 4]
Dev_3 = [0, 1, 1, 1, 2, 2, 3, 3, 4]
labels = ["Dev_1","Dev_2","Dev_3"]
```

```
colors = ['#6d904f', '#fc4f30', '#008fd5']

#want to see which developer work minutes over the days .
plt.stackplot(days,Dev_1,Dev_2,Dev_3,labels = labels,colors = colors)
plt.legend()
plt.title("Project Working")
plt.xlabel("Days")
plt.ylabel("Minutes")
plt.show()
```



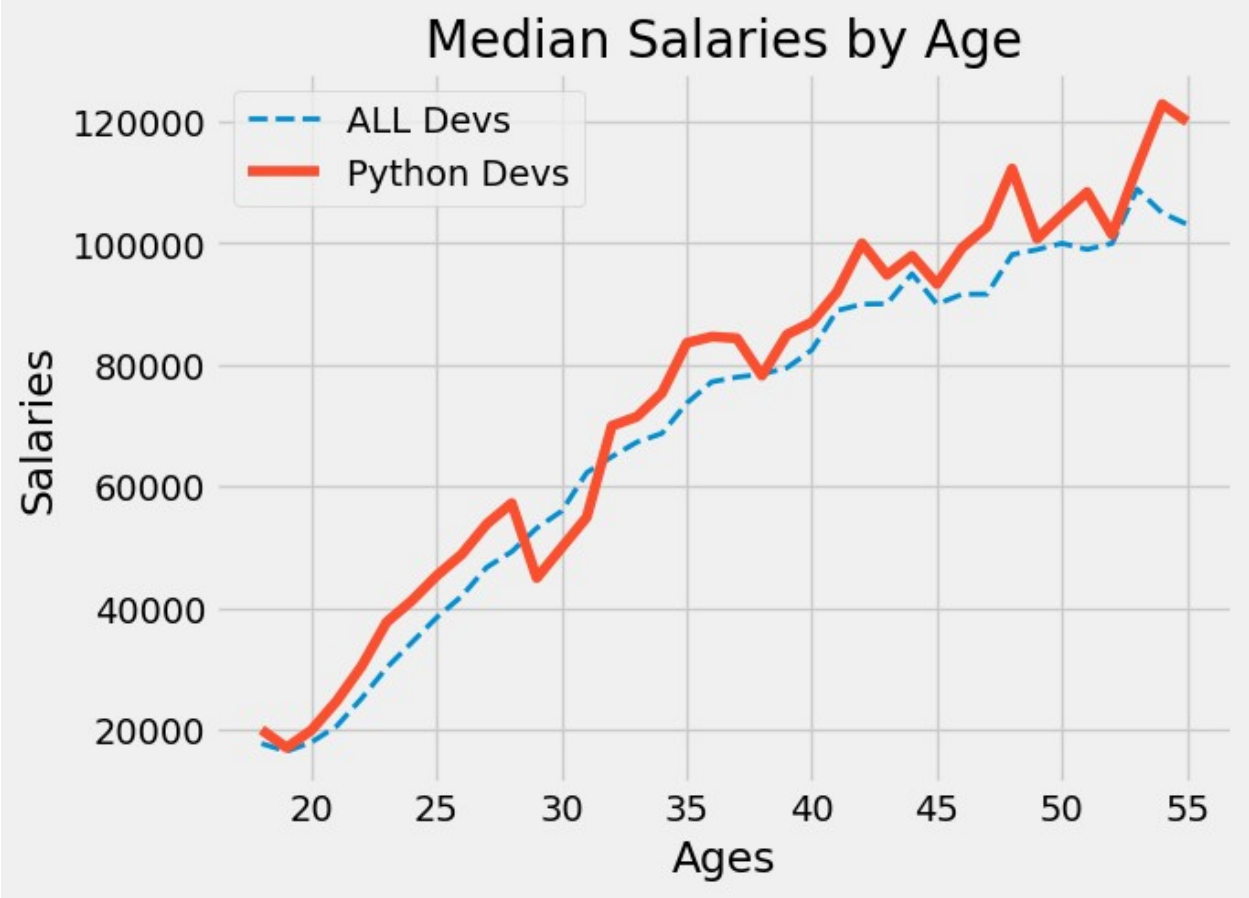
Fill line plot

```
df = pd.read_csv("G:\Self\Python\Matplotlib\dataset\data2.csv")
df
```

	Age	All_Devs	Python	JavaScript
0	18	17784	20046	16446
1	19	16500	17100	16791
2	20	18012	20000	18942
3	21	20628	24744	21780
4	22	25206	30500	25704

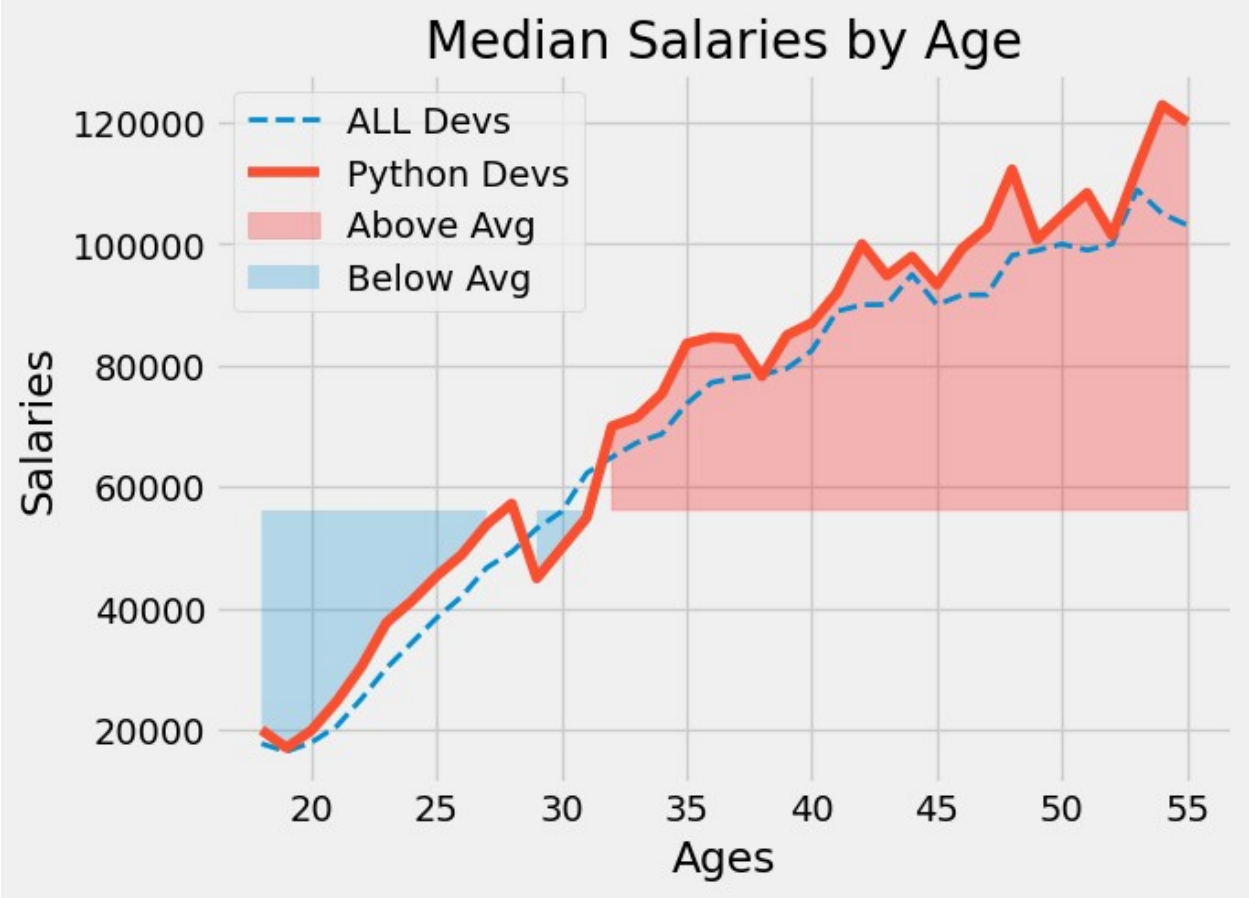
```
5    23     30252   37732        29000
6    24     34368   41247        34372
7    25     38496   45372        37810
8    26     42000   48876        43515
9    27     46752   53850        46823
10   28     49320   57287        49293
11   29     53200   45000        53437
12   30     56000   50000        56373
13   31     62316   55000        62375
14   32     64928   70000        66674
15   33     67317   71496        68745
16   34     68748   75370        68746
17   35     73752   83640        74583
18   36     77232   84666        79000
19   37     78000   84392        78508
20   38     78508   78254        79996
21   39     79536   85000        80403
22   40     82488   87038        83820
23   41     88935   91991        88833
24   42     90000  100000        91660
25   43     90056   94796        87892
26   44     95000   97962        96243
27   45     90000   93302        90000
28   46     91633   99240        99313
29   47     91660  102736        91660
30   48     98150  112285       102264
31   49     98964  100771       100000
32   50    100000  104708       100000
33   51     98988  108423        91660
34   52    100000  101407        99240
35   53    108923  112542       108000
36   54    105000  122870       105000
37   55    103117  120000       104000
```

```
ages = df["Age"]
salaries_All_Devs = df["All_Devs"]
salaries_Python = df["Python"]
#salaries_JavaScript = ["JavaScript"]

plt.plot(ages,salaries_All_Devs,linestyle="--",lw=2,label = "ALL
Devs")
plt.plot(ages,salaries_Python,label = "Python Devs")
plt.legend()
#plt.plot(ages,salaries_JavaScript)

plt.title("Median Salaries by Age")
plt.xlabel("Ages")
plt.ylabel("Salaries")

plt.show()
```



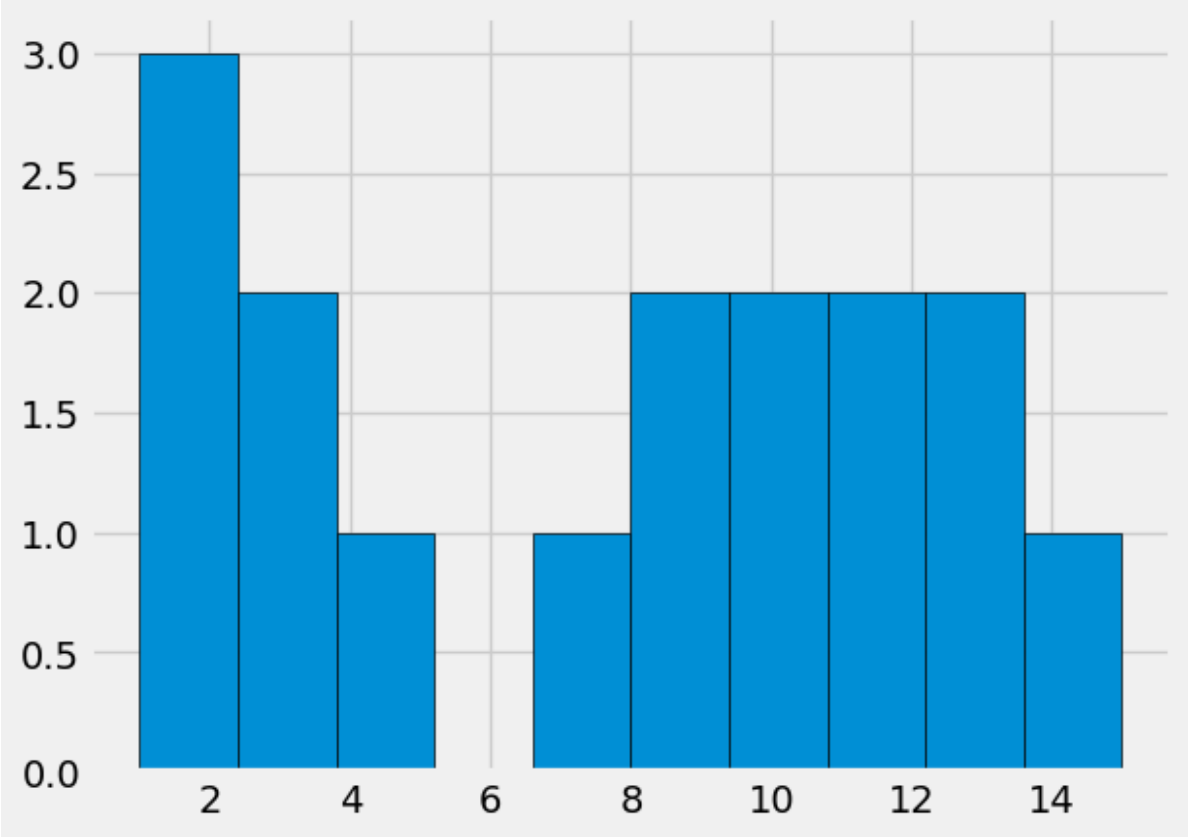

```
median = np.median(df)
#py_median = np.median(salaries_Python)
plt.plot(ages,salaries_All_Devs,linestyle="--",lw=2,label = "ALL
Devs")
plt.plot(ages,salaries_Python,label = "Python Devs")
plt.fill_between(ages,salaries_Python,median,color='red',alpha =
0.25,where=(salaries_Python>median),
                 label = "Above Avg")
plt.fill_between(ages,salaries_Python,median,alpha =
0.25,where=(salaries_Python<median),
                 label = "Below Avg")
#plt.fill_between(ages,salaries_Python,median,alpha =
0.25,color='g',where=(salaries_Python>salaries_All_Devs

#  ),
                 #label = "High py")
plt.legend()
#plt.plot(ages,salaries_JavaScript)

plt.title("Median Salaries by Age")
plt.xlabel("Ages")
plt.ylabel("Salaries")
```

```
plt.show()
```



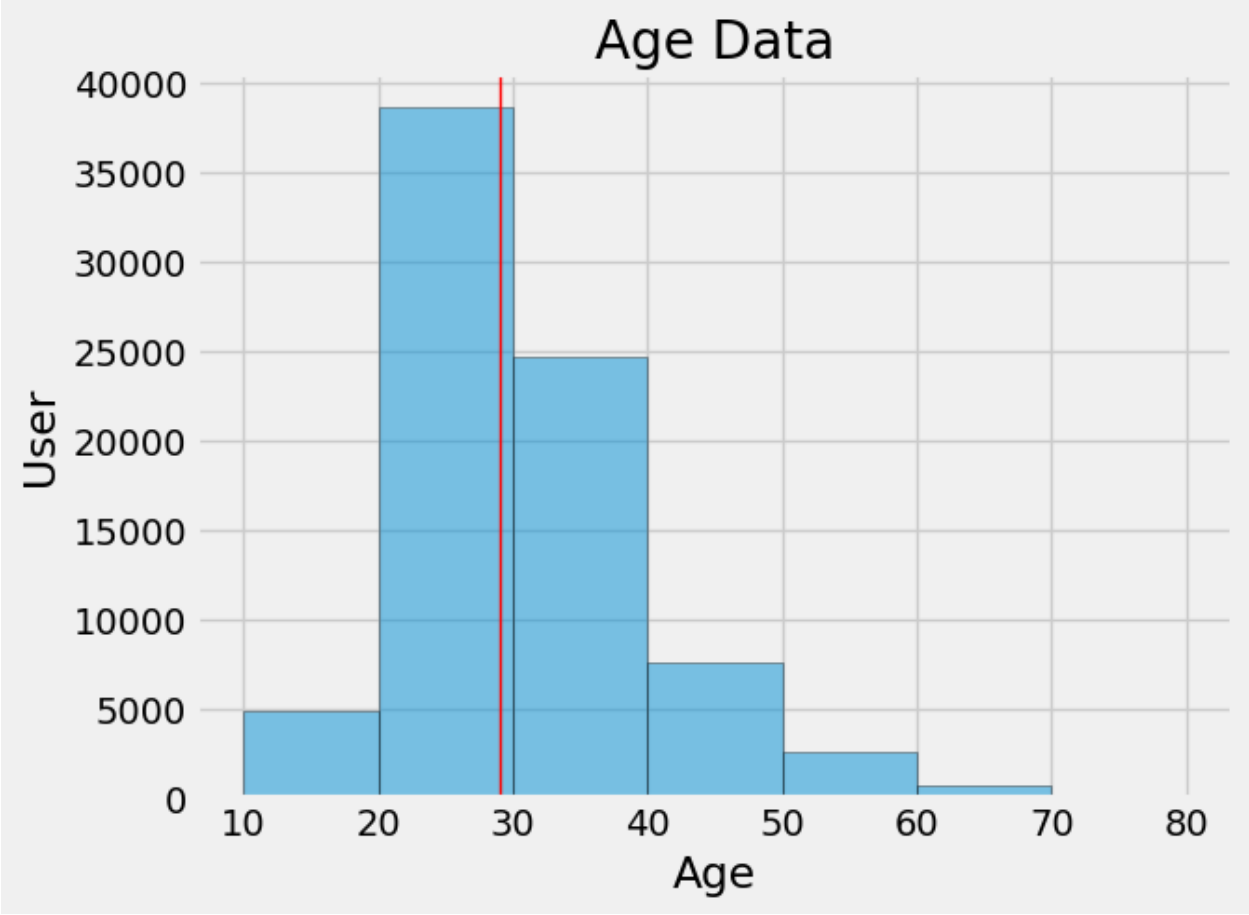
Histogram

```
x = [1,1,2,3,3,5,7,8,9,10,
     10,11,11,13,13,15
     ]

plt.hist(x, bins=10,edgecolor='black')
plt.show()
```



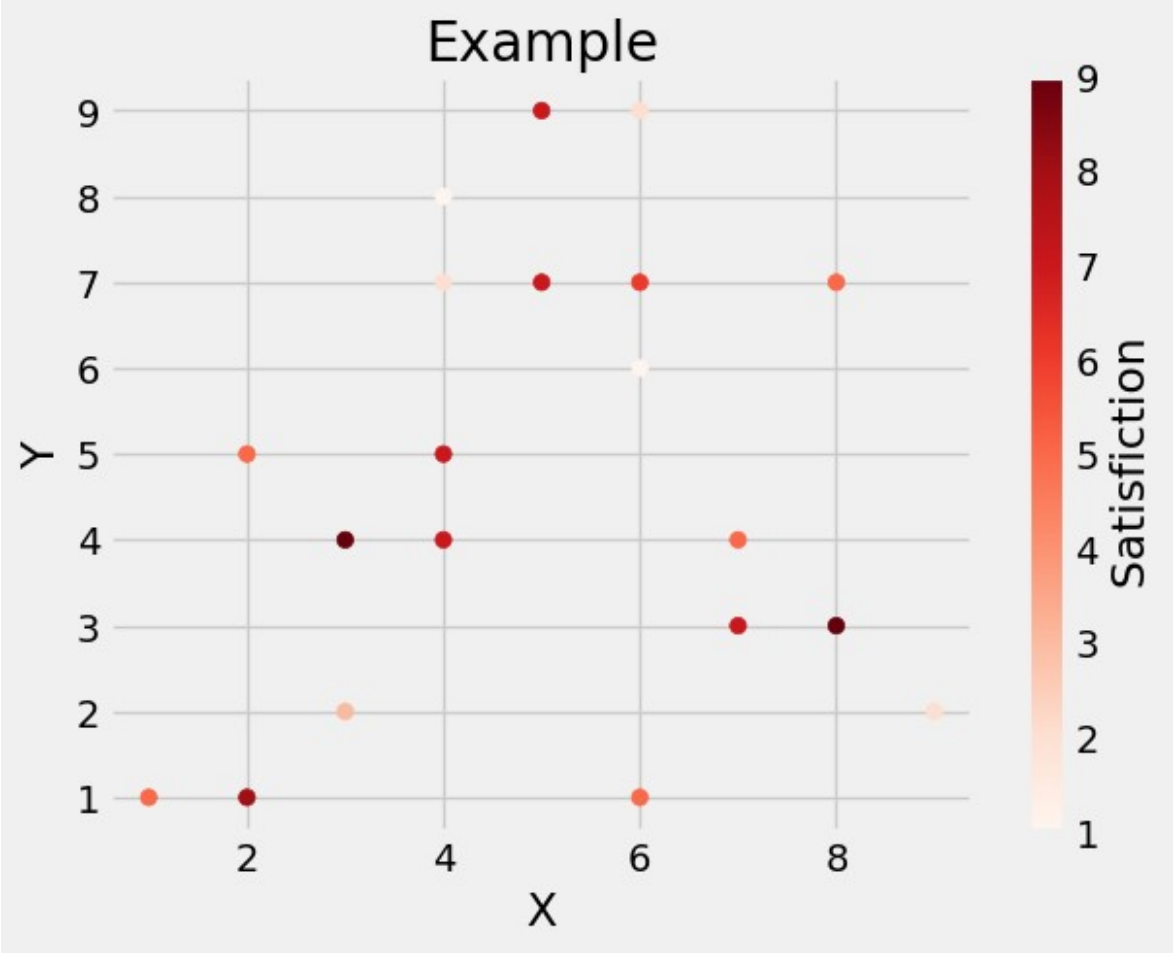

```
#Real data
data = pd.read_csv("G:\Self\Python\Matplotlib\dataset\
data_HistoGram.csv")
data
res = data["Responder_id"]
age = data["Age"]
median = np.median(data["Age"])
bins =[x for x in range(10,90,10)]
plt.hist(age,bins = bins,edgecolor = "black",alpha=.50)#,log=True)
plt.axvline(median,color = "red",label="Age Median",linewidth=1)
plt.title("Age Data")
plt.xlabel("Age")
plt.ylabel("User")
bins
```

```
[10, 20, 30, 40, 50, 60, 70, 80]
```

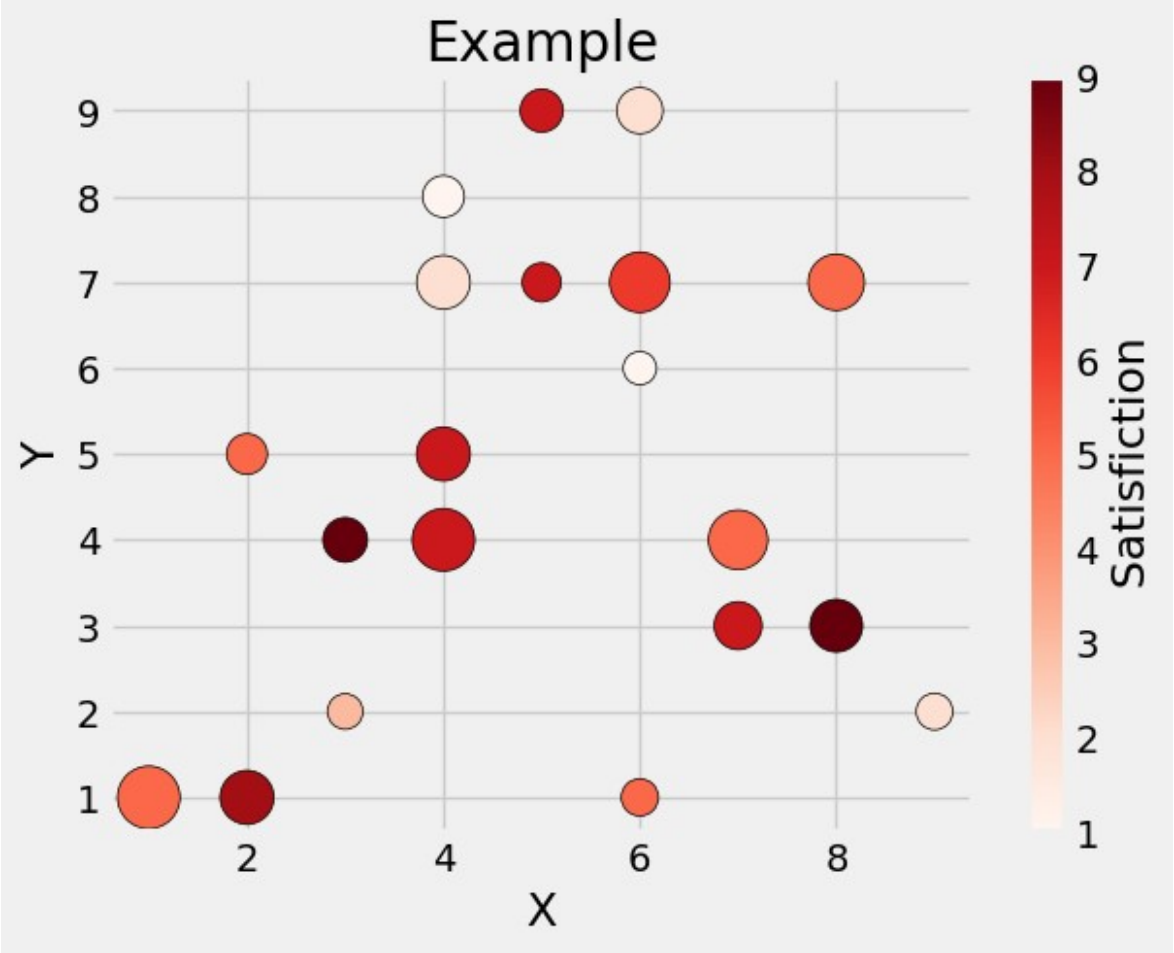


Scatter Plot

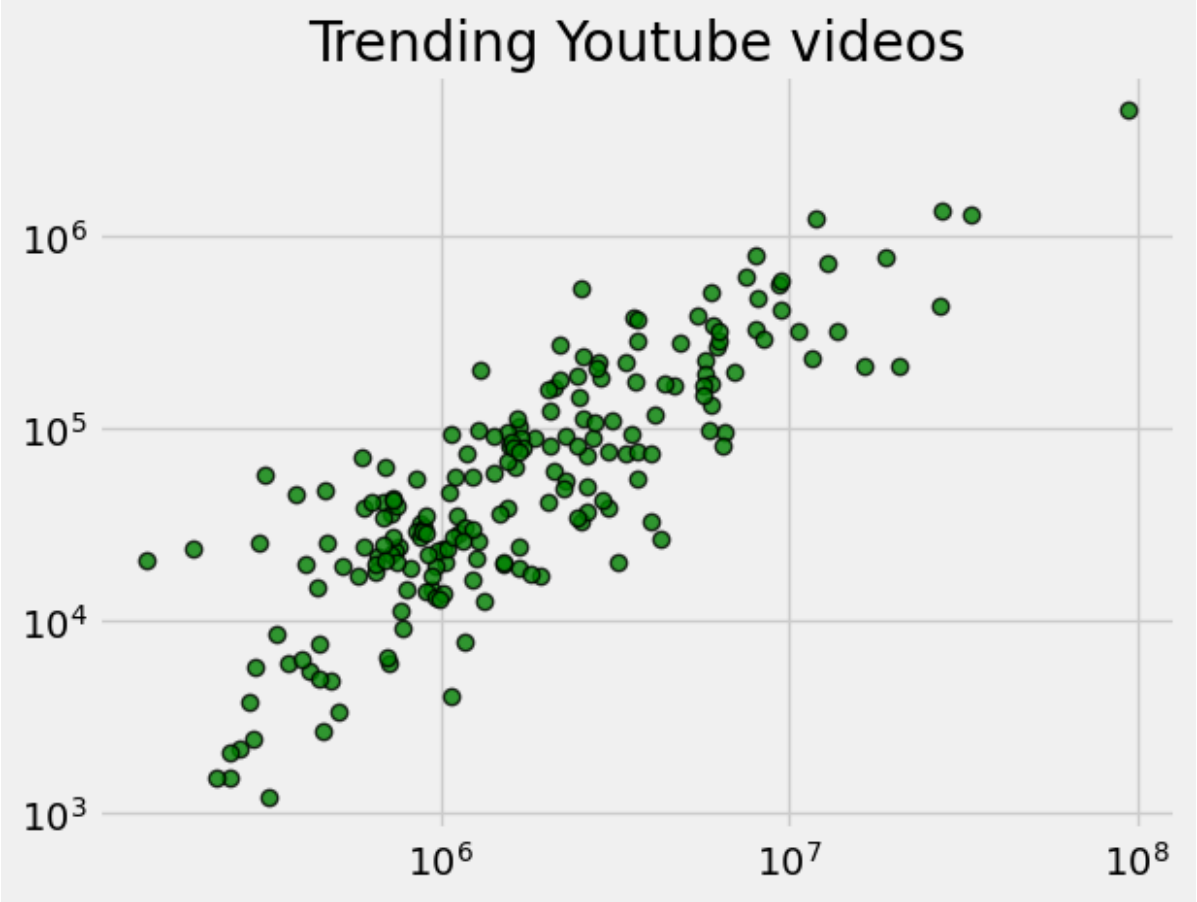
```
x = [5, 7, 8, 5, 6, 7, 9, 2, 3, 4, 4, 4, 2, 6, 3, 6, 8, 6, 4, 1]
y = [7, 4, 3, 9, 1, 3, 2, 5, 2, 4, 8, 7, 1, 6, 4, 9, 7, 7, 5, 1]
Sat_Level = [7, 5, 9, 7, 5, 7, 2, 5, 3, 7, 1, 2, 8, 1, 9, 2, 5, 6, 7,
5]
#plt.scatter(x,y,color = 'green',marker = "X")
plt.scatter(x,y,c = Sat_Level,cmap = "Reds",alpha = 1)
cbar = plt.colorbar()
cbar.set_label("Satisfiction ")
plt.title("Example")
plt.xlabel ("X")
plt.ylabel ("Y")
plt.show()
```




```
x = [5, 7, 8, 5, 6, 7, 9, 2, 3, 4, 4, 4, 2, 6, 3, 6, 8, 6, 4, 1]
y = [7, 4, 3, 9, 1, 3, 2, 5, 2, 4, 8, 7, 1, 6, 4, 9, 7, 7, 5, 1]
Sat_Level = [7, 5, 9, 7, 5, 7, 2, 5, 3, 7, 1, 2, 8, 1, 9, 2, 5, 6, 7,
5]
sizes = [209, 486, 381, 255, 191, 315, 185, 228, 174,
         538, 239, 394, 399, 153, 273, 293, 436, 501, 397, 539]
#plt.scatter(x,y,color = 'green',marker = "X")
plt.scatter(x,y,s=sizes,c = Sat_Level,cmap = "Reds",alpha =
1,edgecolor="black")
cbar = plt.colorbar()
cbar.set_label("Satisfiction ")
plt.title("Example")
plt.xlabel ("X")
plt.ylabel ("Y")
plt.show()
```




```
#Working with Real data

data = pd.read_csv("G:\Self\Python\Matplotlib\dataset\
data_scatter.csv")
view = data["view_count"]
likes = data["likes"]
ratio = data["ratio"]

plt.scatter(view,likes,c = "green",edgecolor = "black",alpha=0.80,lw =
1)
plt.xscale('log')
plt.yscale('log')
plt.title("Trending Youtube videos")
plt.xlabel = "Views"
plt.ylabel = "Likes"
plt.show()
```


Trending Youtube videos
10^6
10^5
10^4
10^3
10^6
10^7
10^8