In [ ]:

```
```

In [ ]:

```
```

Auto MPG

In [1]:

```
#Importing Packages
import numpy as np
import pandas as pd
```

In [2]:

```
#Reading the Dataset
auto = pd.read_csv(r"C:\Users\VICTUS\Downloads\auto+mpg\auto-mpg.data",header=None,delim_whitespace=True)
auto
```

```
C:\Users\VICTUS\AppData\Local\Temp\ipykernel_23848\4017178933.py:2: FutureWarning: The 'delim_whitespace' keyword in pd.read_cs
v is deprecated and will be removed in a future version. Use ``sep='\s+'`` instead
  auto = pd.read_csv(r"C:\Users\VICTUS\Downloads\auto+mpg\auto-mpg.data",header=None,delim_whitespace=True)
```

Out[2]:

	0	1	2	3	4	5	6	7	8
0	18.0	8	307.0	130.0	3504.0	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693.0	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436.0	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433.0	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449.0	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86.00	2790.0	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52.00	2130.0	24.6	82	2	vw pickup
395	32.0	4	135.0	84.00	2295.0	11.6	82	1	dodge rampage
396	28.0	4	120.0	79.00	2625.0	18.6	82	1	ford ranger
397	31.0	4	119.0	82.00	2720.0	19.4	82	1	chevy s-10

398 rows × 9 columns

In [3]:

```
auto = pd.read_csv(r"C:\Users\VICTUS\Downloads\auto+mpg\auto-mpg.data",header=None,sep='\s+')
auto
```

```
<>:1: SyntaxWarning: invalid escape sequence '\s'
<>:1: SyntaxWarning: invalid escape sequence '\s'
C:\Users\VICTUS\AppData\Local\Temp\ipykernel_23848\551619158.py:1: SyntaxWarning: invalid escape sequence '\s'
  auto = pd.read_csv(r"C:\Users\VICTUS\Downloads\auto+mpg\auto-mpg.data",header=None,sep='\s+')
```

Out[3]:

	0	1	2	3	4	5	6	7	8
0	18.0	8	307.0	130.0	3504.0	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693.0	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436.0	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433.0	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449.0	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86.00	2790.0	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52.00	2130.0	24.6	82	2	vw pickup
395	32.0	4	135.0	84.00	2295.0	11.6	82	1	dodge rampage
396	28.0	4	120.0	79.00	2625.0	18.6	82	1	ford ranger
397	31.0	4	119.0	82.00	2720.0	19.4	82	1	chevy s-10

398 rows × 9 columns

In [4]:

```
#Rename the columns (if required)
auto.columns = ['mpg','cylinders','displacement','horsepower',
                'weight','acceleration','model year','origin',
                'car name']
auto.head()
```

Out[4]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130.0	3504.0	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693.0	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436.0	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433.0	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449.0	10.5	70	1	ford torino

In [5]:

```
#check missing values
auto.isnull().sum()
```

Out[5]:

```
mpg             0
cylinders       0
displacement    0
horsepower      0
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

In [6]:

```
#check data types
auto.dtypes
```

Out[6]:

```
mpg             float64
cylinders         int64
displacement    float64
horsepower       object
weight          float64
acceleration    float64
model year        int64
origin            int64
car name         object
dtype: object
```

In [7]:
```
#check unique values
for i in auto.columns:
    print(i,':','\n',auto[i].unique(),'\n')
```

```
mpg :
 [18.  15.  16.  17.  14.  24.  22.  21.  27.  26.  25.  10.  11.   9.
 28.  19.  12.  13.  23.  30.  31.  35.  20.  29.  32.  33.  17.5 15.5
 14.5 22.5 24.5 18.5 29.5 26.5 16.5 31.5 36.  25.5 33.5 20.5 30.5 21.5
 43.1 36.1 32.8 39.4 19.9 19.4 20.2 19.2 25.1 20.6 20.8 18.6 18.1 17.7
 27.5 27.2 30.9 21.1 23.2 23.8 23.9 20.3 21.6 16.2 19.8 22.3 17.6 18.2
 16.9 31.9 34.1 35.7 27.4 25.4 34.2 34.5 31.8 37.3 28.4 28.8 26.8 41.5
 38.1 32.1 37.2 26.4 24.3 19.1 34.3 29.8 31.3 37.  32.2 46.6 27.9 40.8
 44.3 43.4 36.4 44.6 40.9 33.8 32.7 23.7 23.6 32.4 26.6 25.8 23.5 39.1
 39.  35.1 32.3 37.7 34.7 34.4 29.9 33.7 32.9 31.6 28.1 30.7 24.2 22.4
 34.  38.  44. ]

cylinders :
 [8 4 6 3 5]

displacement :
 [307.  350.  318.  304.  302.  429.  454.  440.  455.  390.  383.  340.
 400.  113.  198.  199.  200.   97.  110.  107.  104.  121.  360.  140.
  98.  232.  225.  250.  351.  258.  122.  116.   79.   88.   71.   72.
  91.   97.5  70.  120.   96.  108.  155.   68.  114.  156.   76.   83.
  90.  231.  262.  134.  119.  171.  115.  101.  305.   85.  130.  168.
 111.  260.  151.  146.   80.   78.  105.  131.  163.   89.  267.   86.
 183.  141.  173.  135.   81.  100.  145.  112.  181.  144. ]

horsepower :
 ['130.0' '165.0' '150.0' '140.0' '198.0' '220.0' '215.0' '225.0' '190.0'
 '170.0' '160.0' '95.00' '97.00' '85.00' '88.00' '46.00' '87.00' '90.00'
 '113.0' '200.0' '210.0' '193.0' '?' '100.0' '105.0' '175.0' '153.0'
 '180.0' '110.0' '72.00' '86.00' '70.00' '76.00' '65.00' '69.00' '60.00'
 '80.00' '54.00' '208.0' '155.0' '112.0' '92.00' '145.0' '137.0' '158.0'
 '167.0' '94.00' '107.0' '230.0' '49.00' '75.00' '91.00' '122.0' '67.00'
 '83.00' '78.00' '52.00' '61.00' '93.00' '148.0' '129.0' '96.00' '71.00'
 '98.00' '115.0' '53.00' '81.00' '79.00' '120.0' '152.0' '102.0' '108.0'
 '68.00' '58.00' '149.0' '89.00' '63.00' '48.00' '66.00' '139.0' '103.0'
 '125.0' '133.0' '138.0' '135.0' '142.0' '77.00' '62.00' '132.0' '84.00'
 '64.00' '74.00' '116.0' '82.00']

weight :
 [3504. 3693. 3436. 3433. 3449. 4341. 4354. 4312. 4425. 3850. 3563. 3609.
 3761. 3086. 2372. 2833. 2774. 2587. 2130. 1835. 2672. 2430. 2375. 2234.
 2648. 4615. 4376. 4382. 4732. 2264. 2228. 2046. 2634. 3439. 3329. 3302.
 3288. 4209. 4464. 4154. 4096. 4955. 4746. 5140. 2962. 2408. 3282. 3139.
 2220. 2123. 2074. 2065. 1773. 1613. 1834. 1955. 2278. 2126. 2254. 2226.
 4274. 4385. 4135. 4129. 3672. 4633. 4502. 4456. 4422. 2330. 3892. 4098.
 4294. 4077. 2933. 2511. 2979. 2189. 2395. 2288. 2506. 2164. 2100. 4100.
 3988. 4042. 3777. 4952. 4363. 4237. 4735. 4951. 3821. 3121. 3278. 2945.
 3021. 2904. 1950. 4997. 4906. 4654. 4499. 2789. 2279. 2401. 2379. 2124.
 2310. 2472. 2265. 4082. 4278. 1867. 2158. 2582. 2868. 3399. 2660. 2807.
 3664. 3102. 2875. 2901. 3336. 2451. 1836. 2542. 3781. 3632. 3613. 4141.
 4699. 4457. 4638. 4257. 2219. 1963. 2300. 1649. 2003. 2125. 2108. 2246.
 2489. 2391. 2000. 3264. 3459. 3432. 3158. 4668. 4440. 4498. 4657. 3907.
 3897. 3730. 3785. 3039. 3221. 3169. 2171. 2639. 2914. 2592. 2702. 2223.
 2545. 2984. 1937. 3211. 2694. 2957. 2671. 1795. 2464. 2572. 2255. 2202.
 4215. 4190. 3962. 3233. 3353. 3012. 3085. 2035. 3651. 3574. 3645. 3193.
 1825. 1990. 2155. 2565. 3150. 3940. 3270. 2930. 3820. 4380. 4055. 3870.
 3755. 2045. 1945. 3880. 4060. 4140. 4295. 3520. 3425. 3630. 3525. 4220.
 4165. 4325. 4335. 1940. 2740. 2755. 2051. 2075. 1985. 2190. 2815. 2600.
 2720. 1800. 2070. 3365. 3735. 3570. 3535. 3155. 2965. 3430. 3210. 3380.
 3070. 3620. 3410. 3445. 3205. 4080. 2560. 2230. 2515. 2745. 2855. 2405.
 2830. 3140. 2795. 2135. 3245. 2990. 2890. 3265. 3360. 3840. 3725. 3955.
 3830. 4360. 4054. 3605. 1925. 1975. 1915. 2670. 3530. 3900. 3190. 3420.
 2200. 2150. 2020. 2595. 2700. 2556. 2144. 1968. 2120. 2019. 2678. 2870.
 3003. 3381. 2188. 2711. 2434. 2110. 2800. 2085. 2335. 2950. 3250. 1850.
 2145. 1845. 2910. 2420. 2500. 2905. 2290. 2490. 2635. 2620. 2725. 2385.
 1755. 1875. 1760. 2050. 2215. 2380. 2320. 2210. 2350. 2615. 3230. 3160.
 2900. 3415. 3060. 3465. 2605. 2640. 2575. 2525. 2735. 2865. 3035. 1980.
 2025. 1970. 2160. 2205. 2245. 1965. 1995. 3015. 2585. 2835. 2665. 2370.
 2790. 2295. 2625.]

acceleration :
 [12.  11.5 11.  10.5 10.   9.   8.5  8.   9.5 15.  15.5 16.  14.5 20.5
 17.5 12.5 14.  13.5 18.5 19.  13.  19.5 18.  17.  23.5 16.5 21.  16.9
 14.9 17.7 15.3 13.9 12.8 15.4 17.6 22.2 22.1 14.2 17.4 16.2 17.8 12.2
 16.4 13.6 15.7 13.2 21.9 16.7 12.1 14.8 18.6 16.8 13.7 11.1 11.4 18.2
 15.8 15.9 14.1 21.5 14.4 19.4 19.2 17.2 18.7 15.1 13.4 11.2 14.7 16.6
 17.3 15.2 14.3 20.1 24.8 11.3 12.9 18.8 18.1 17.9 21.7 23.7 19.9 21.8
 13.8 12.6 16.1 20.7 18.3 20.4 19.6 17.1 15.6 24.6 11.6]

model year :
 [70 71 72 73 74 75 76 77 78 79 80 81 82]

origin :
 [1 3 2]

car name :
 ['chevrolet chevelle malibu' 'buick skylark 320' 'plymouth satellite'
 'amc rebel sst' 'ford torino' 'ford galaxie 500' 'chevrolet impala'
 'plymouth fury iii' 'pontiac catalina' 'amc ambassador dpl'
 'dodge challenger se' "plymouth 'cuda 340" 'chevrolet monte carlo'
 'buick estate wagon (sw)' 'toyota corona mark ii' 'plymouth duster'
 'amc hornet' 'ford maverick' 'datsun pl510'
 'volkswagen 1131 deluxe sedan' 'peugeot 504' 'audi 100 ls' 'saab 99e'
 'bmw 2002' 'amc gremlin' 'ford f250' 'chevy c20' 'dodge d200' 'hi 1200d'
 'chevrolet vega 2300' 'toyota corona' 'ford pinto'
 'plymouth satellite custom' 'ford torino 500' 'amc matador'
 'pontiac catalina brougham' 'dodge monaco (sw)'
 'ford country squire (sw)' 'pontiac safari (sw)'
 'amc hornet sportabout (sw)' 'chevrolet vega (sw)' 'pontiac firebird'
 'ford mustang' 'mercury capri 2000' 'opel 1900' 'peugeot 304' 'fiat 124b'
 'toyota corolla 1200' 'datsun 1200' 'volkswagen model 111'
 'plymouth cricket' 'toyota corona hardtop' 'dodge colt hardtop'
 'volkswagen type 3' 'chevrolet vega' 'ford pinto runabout'
 'amc ambassador sst' 'mercury marquis' 'buick lesabre custom'
 'oldsmobile delta 88 royale' 'chrysler newport royal' 'mazda rx2 coupe'
 'amc matador (sw)' 'chevrolet chevelle concours (sw)'
 'ford gran torino (sw)' 'plymouth satellite custom (sw)'
 'volvo 145e (sw)' 'volkswagen 411 (sw)' 'peugeot 504 (sw)'
 'renault 12 (sw)' 'ford pinto (sw)' 'datsun 510 (sw)'
 'toyouta corona mark ii (sw)' 'dodge colt (sw)'
 'toyota corolla 1600 (sw)' 'buick century 350' 'chevrolet malibu'
 'ford gran torino' 'dodge coronet custom' 'mercury marquis brougham'
 'chevrolet caprice classic' 'ford ltd' 'plymouth fury gran sedan'
 'chrysler new yorker brougham' 'buick electra 225 custom'
 'amc ambassador brougham' 'plymouth valiant' 'chevrolet nova custom'
 'volkswagen super beetle' 'ford country' 'plymouth custom suburb'
 'oldsmobile vista cruiser' 'toyota carina' 'datsun 610' 'maxda rx3'
 'mercury capri v6' 'fiat 124 sport coupe' 'chevrolet monte carlo s'
 'pontiac grand prix' 'fiat 128' 'opel manta' 'audi 100ls' 'volvo 144ea'
 'dodge dart custom' 'saab 99le' 'toyota mark ii' 'oldsmobile omega'
 'chevrolet nova' 'datsun b210' 'chevrolet chevelle malibu classic'
 'plymouth satellite sebring' 'buick century luxus (sw)'
 'dodge coronet custom (sw)' 'audi fox' 'volkswagen dasher' 'datsun 710'
 'dodge colt' 'fiat 124 tc' 'honda civic' 'subaru' 'fiat x1.9'
 'plymouth valiant custom' 'mercury monarch' 'chevrolet bel air'
 'plymouth grand fury' 'buick century' 'chevroelt chevelle malibu'
 'plymouth fury' 'buick skyhawk' 'chevrolet monza 2+2' 'ford mustang ii'
 'toyota corolla' 'pontiac astro' 'volkswagen rabbit' 'amc pacer'
 'volvo 244dl' 'honda civic cvcc' 'fiat 131' 'capri ii' 'renault 12tl'
 'dodge coronet brougham' 'chevrolet chevette' 'chevrolet woody'
 'vw rabbit' 'dodge aspen se' 'ford granada ghia' 'pontiac ventura sj'
 'amc pacer d/l' 'datsun b-210' 'volvo 245' 'plymouth volare premier v8'
 'mercedes-benz 280s' 'cadillac seville' 'chevy c10' 'ford f108'
 'dodge d100' 'honda accord cvcc' 'buick opel isuzu deluxe'
 'renault 5 gtl' 'plymouth arrow gs' 'datsun f-10 hatchback'
 'oldsmobile cutlass supreme' 'dodge monaco brougham'
 'mercury cougar brougham' 'chevrolet concours' 'buick skylark'
 'plymouth volare custom' 'ford granada' 'pontiac grand prix lj'
 'chevrolet monte carlo landau' 'chrysler cordoba' 'ford thunderbird'
 'volkswagen rabbit custom' 'pontiac sunbird coupe'
 'toyota corolla liftback' 'ford mustang ii 2+2' 'dodge colt m/m'
 'subaru dl' 'datsun 810' 'bmw 320i' 'mazda rx-4'
 'volkswagen rabbit custom diesel' 'ford fiesta' 'mazda glc deluxe'
 'datsun b210 gx' 'oldsmobile cutlass salon brougham' 'dodge diplomat'
 'mercury monarch ghia' 'pontiac phoenix lj' 'ford fairmont (auto)'
 'ford fairmont (man)' 'plymouth volare' 'amc concord'
 'buick century special' 'mercury zephyr' 'dodge aspen' 'amc concord d/l'
 'buick regal sport coupe (turbo)' 'ford futura' 'dodge magnum xe'
 'datsun 510' 'dodge omni' 'toyota celica gt liftback' 'plymouth sapporo'
 'oldsmobile starfire sx' 'datsun 200-sx' 'audi 5000' 'volvo 264gl'
 'saab 99gle' 'peugeot 604sl' 'volkswagen scirocco' 'honda accord lx'
 'pontiac lemans v6' 'mercury zephyr 6' 'ford fairmont 4'
 'amc concord dl 6' 'dodge aspen 6' 'ford ltd landau'
 'mercury grand marquis' 'dodge st. regis' 'chevrolet malibu classic (sw)'
 'chrysler lebaron town @ country (sw)' 'vw rabbit custom'
 'maxda glc deluxe' 'dodge colt hatchback custom' 'amc spirit dl'
 'mercedes benz 300d' 'cadillac eldorado' 'plymouth horizon'
 'plymouth horizon tc3' 'datsun 210' 'fiat strada custom'
 'buick skylark limited' 'chevrolet citation' 'oldsmobile omega brougham'
 'pontiac phoenix' 'toyota corolla tercel' 'datsun 310' 'ford fairmont'
 'audi 4000' 'toyota corona liftback' 'mazda 626' 'datsun 510 hatchback'
 'mazda glc' 'vw rabbit c (diesel)' 'vw dasher (diesel)'
 'audi 5000s (diesel)' 'mercedes-benz 240d' 'honda civic 1500 gl'
 'renault lecar deluxe' 'vokswagen rabbit' 'datsun 280-zx' 'mazda rx-7 gs'
 'triumph tr7 coupe' 'ford mustang cobra' 'honda accord'
 'plymouth reliant' 'dodge aries wagon (sw)' 'toyota starlet'
 'plymouth champ' 'honda civic 1300' 'datsun 210 mpg' 'toyota tercel'
 'mazda glc 4' 'plymouth horizon 4' 'ford escort 4w' 'ford escort 2h'
 'volkswagen jetta' 'renault 18i' 'honda prelude' 'datsun 200sx'
 'peugeot 505s turbo diesel' 'volvo diesel' 'toyota cressida'
 'datsun 810 maxima' 'oldsmobile cutlass ls' 'ford granada gl'
 'chrysler lebaron salon' 'chevrolet cavalier' 'chevrolet cavalier wagon'
 'chevrolet cavalier 2-door' 'pontiac j2000 se hatchback' 'dodge aries se'
 'ford fairmont futura' 'amc concord dl' 'volkswagen rabbit l'
 'mazda glc custom l' 'mazda glc custom' 'plymouth horizon miser'
 'mercury lynx l' 'nissan stanza xe' 'honda civic (auto)' 'datsun 310 gx'
 'buick century limited' 'oldsmobile cutlass ciera (diesel)'
 'chrysler lebaron medallion' 'ford granada l' 'toyota celica gt'
 'dodge charger 2.2' 'chevrolet camaro' 'ford mustang gl' 'vw pickup'
 'dodge rampage' 'ford ranger' 'chevy s-10']
```

In [9]:

```
for i in auto.columns:
    print(i,':',sum(auto[i]=='?'))
```

```
mpg : 0
cylinders : 0
displacement : 0
horsepower : 6
weight : 0
acceleration : 0
model year : 0
origin : 0
car name : 0
```

In [10]:

```
d = auto.describe(include='all')
d
```

Out[10]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
count	398.000000	398.000000	398.000000	398	398.000000	398.000000	398.000000	398.000000	398
unique	NaN	NaN	NaN	94	NaN	NaN	NaN	NaN	305
top	NaN	NaN	NaN	150.0	NaN	NaN	NaN	NaN	ford pinto
freq	NaN	NaN	NaN	22	NaN	NaN	NaN	NaN	6
mean	23.514573	5.454774	193.425879	NaN	2970.424623	15.568090	76.010050	1.572864	NaN
std	7.815984	1.701004	104.269838	NaN	846.841774	2.757689	3.697627	0.802055	NaN
min	9.000000	3.000000	68.000000	NaN	1613.000000	8.000000	70.000000	1.000000	NaN
25%	17.500000	4.000000	104.250000	NaN	2223.750000	13.825000	73.000000	1.000000	NaN
50%	23.000000	4.000000	148.500000	NaN	2803.500000	15.500000	76.000000	1.000000	NaN
75%	29.000000	8.000000	262.000000	NaN	3608.000000	17.175000	79.000000	2.000000	NaN
max	46.600000	8.000000	455.000000	NaN	5140.000000	24.800000	82.000000	3.000000	NaN

In [11]:

```
auto.horsepower.replace('?',d['horsepower'][2],inplace=True)
```

```
C:\Users\VICTUS\AppData\Local\Temp\ipykernel_23848\227787081.py:1: FutureWarning: Series.__getitem__ treating keys as positions
is deprecated. In a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To acce
ss a value by position, use `ser.iloc[pos]`
  auto.horsepower.replace('?',d['horsepower'][2],inplace=True)
```

In [12]:

```
for i in auto.columns:
    print(i,':',sum(auto[i]=='?'))
```

```
mpg : 0
cylinders : 0
displacement : 0
horsepower : 0
weight : 0
acceleration : 0
model year : 0
origin : 0
car name : 0
```

In [13]:

```
auto.dtypes
```

Out[13]:

```
mpg             float64
cylinders         int64
displacement    float64
horsepower       object
weight          float64
acceleration    float64
model year        int64
origin            int64
car name         object
dtype: object
```

In [15]:

```
auto.horsepower = auto.horsepower.astype(float)
auto.dtypes
```

Out[15]:

```
mpg             float64
cylinders         int64
displacement    float64
horsepower      float64
weight          float64
acceleration    float64
model year        int64
origin            int64
car name         object
dtype: object
```

Replace '?' with np.nan --> change dtype to float using astype --> find describe and you'll get avg --> replace np.nan with avg of that column

Matplotlib

Data Visualization Technique

Matplotlib is a Python library for data visualization, primarily used to create static, animated and interactive plots.

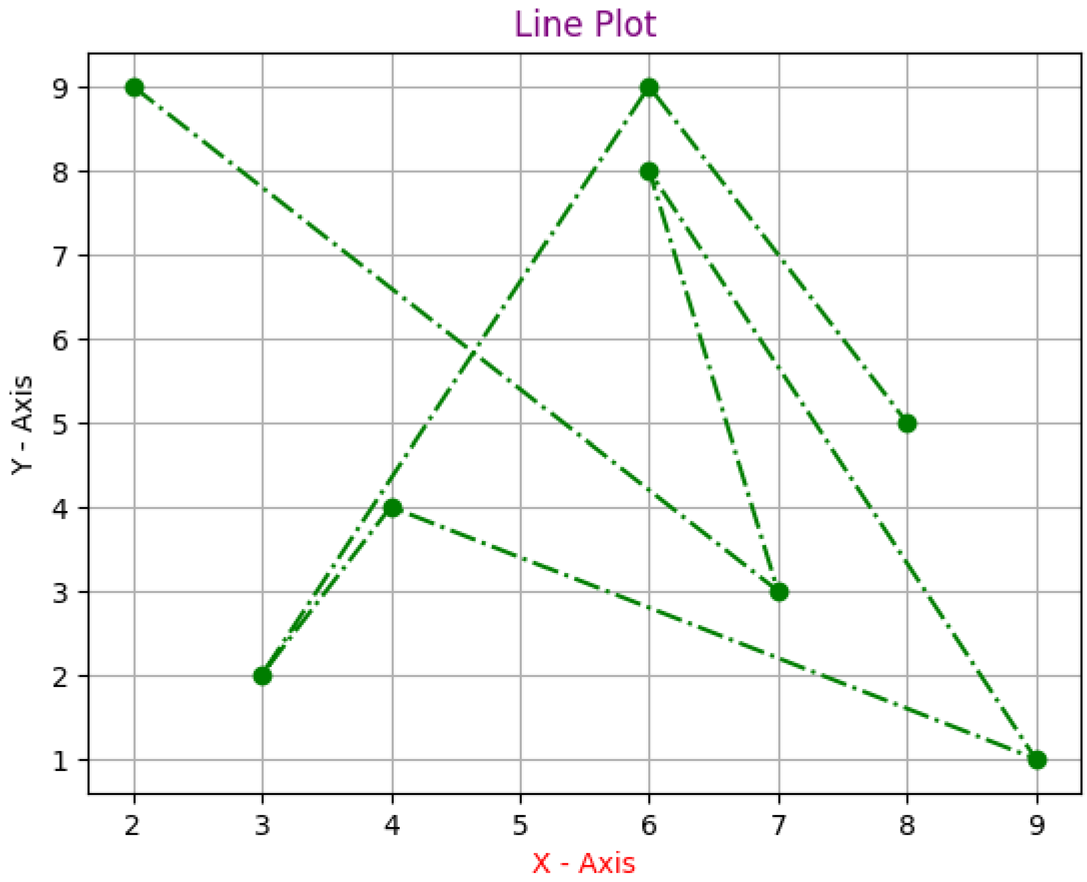
It provides a wide range of plotting functions to visualize data effectively.

In [16]:

```
import matplotlib.pyplot as plt
```

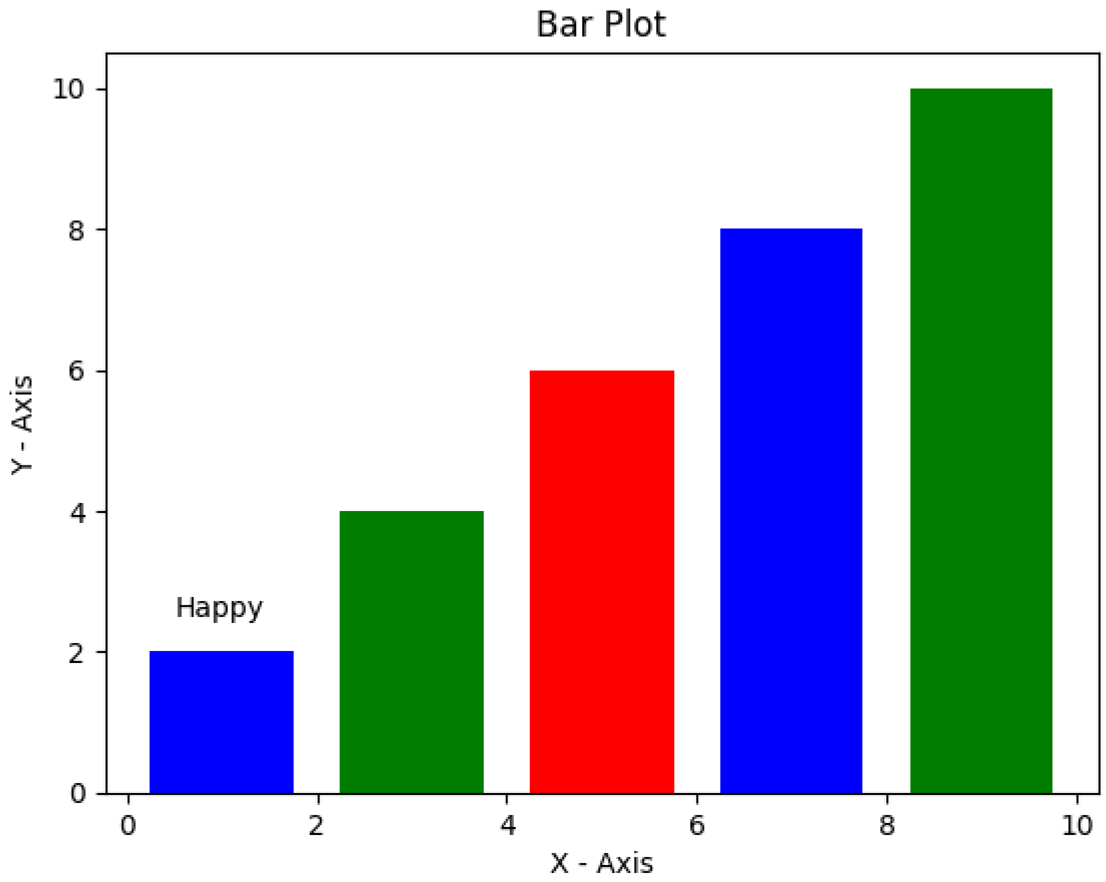
In [38]:

```
#Line Plot
x = [2,7,6,9,4,3,6,8]
y = [9,3,8,1,4,2,9,5]

plt.plot(x,y,'og-.')
plt.title("Line Plot",color='purple')
plt.xlabel('X - Axis',color='r')
plt.ylabel('Y - Axis')
plt.grid()
plt.show()
```



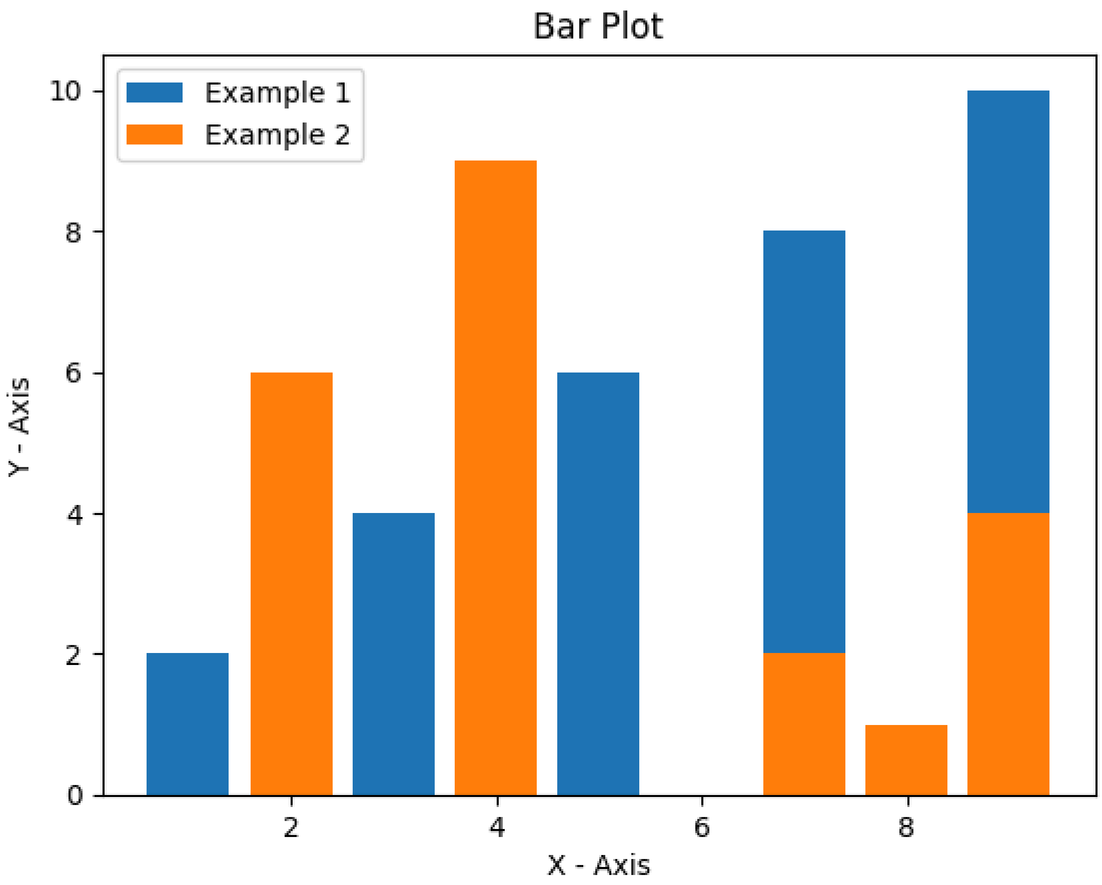
In [51]:

```
#Bar Plot
x = [1,3,5,7,9]
y = [2,4,6,8,10]

plt.bar(x,y,color=['b','g','r'],width=1.5)
plt.title("Bar Plot")
plt.xlabel('X - Axis')
plt.ylabel('Y - Axis')
plt.annotate("Happy",xy=(0.5,2.5))
plt.show()
```



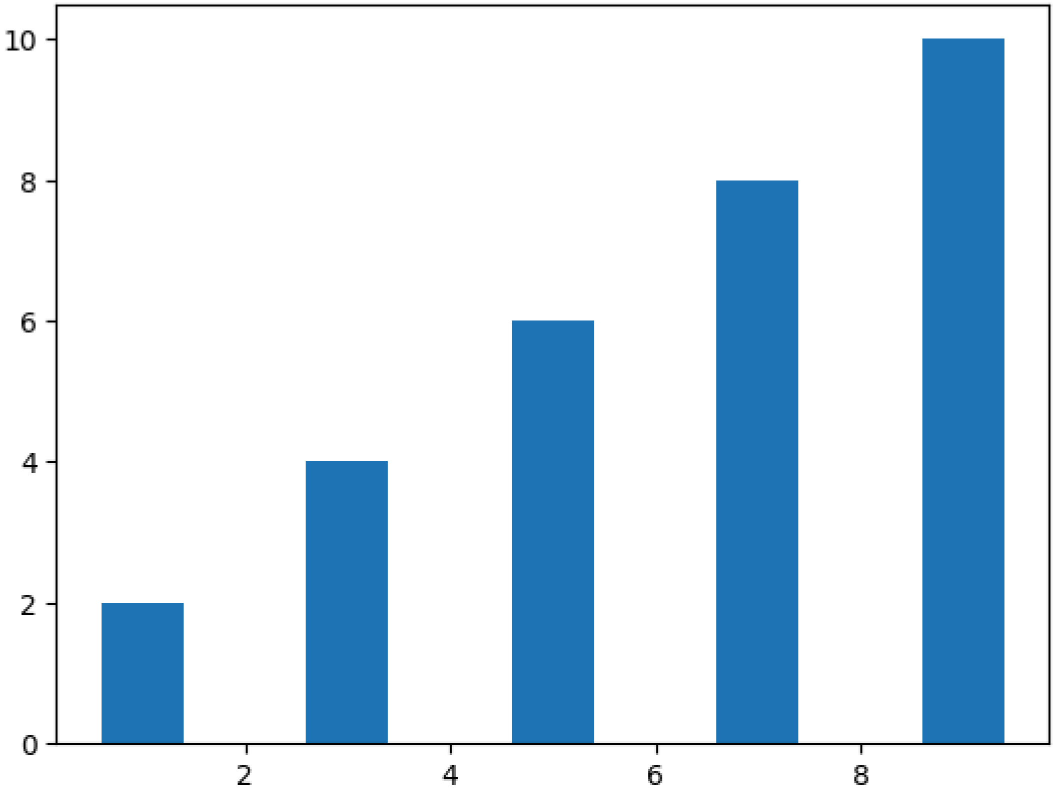
In [53]:

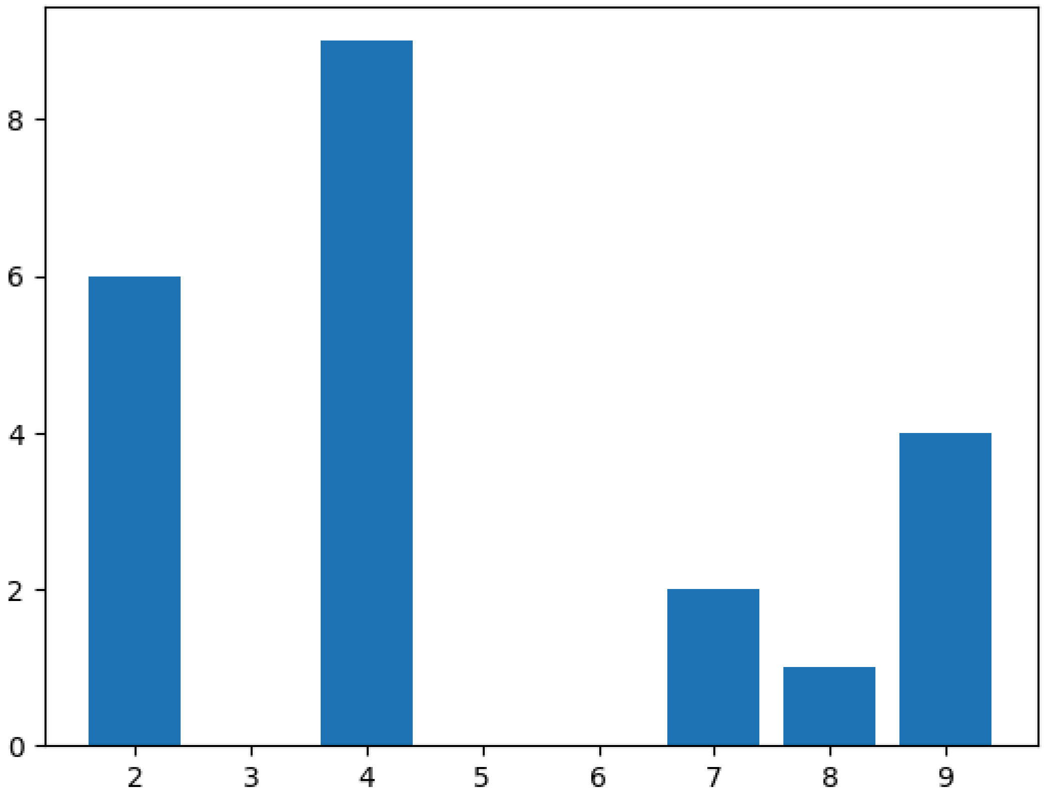
```python
#Multiple Bar Plot
x = [1,3,5,7,9]
y = [2,4,6,8,10]
plt.bar(x,y,label='Example 1')

x1 = [4,7,2,9,8]
y1 = [9,2,6,4,1]
plt.bar(x1,y1,label='Example 2')

plt.title("Bar Plot")
plt.legend()
plt.xlabel('X - Axis')
plt.ylabel('Y - Axis')
plt.show()
```



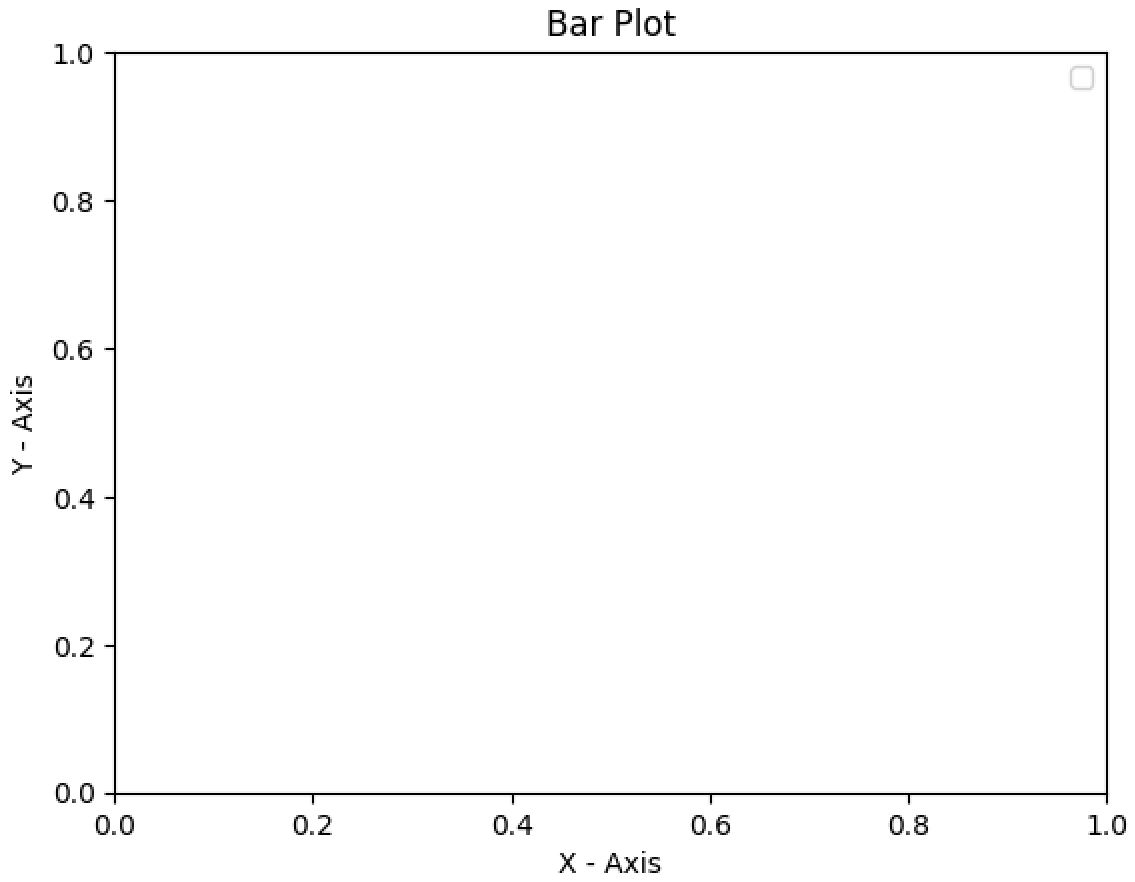
In [54]:

```
#Multiple Bar Plot
x = [1,3,5,7,9]
y = [2,4,6,8,10]
plt.bar(x,y,label='Example 1')
plt.show()

x1 = [4,7,2,9,8]
y1 = [9,2,6,4,1]
plt.bar(x1,y1,label='Example 2')
plt.show()
plt.title("Bar Plot")
plt.legend()
plt.xlabel('X - Axis')
plt.ylabel('Y - Axis')
plt.show()
```

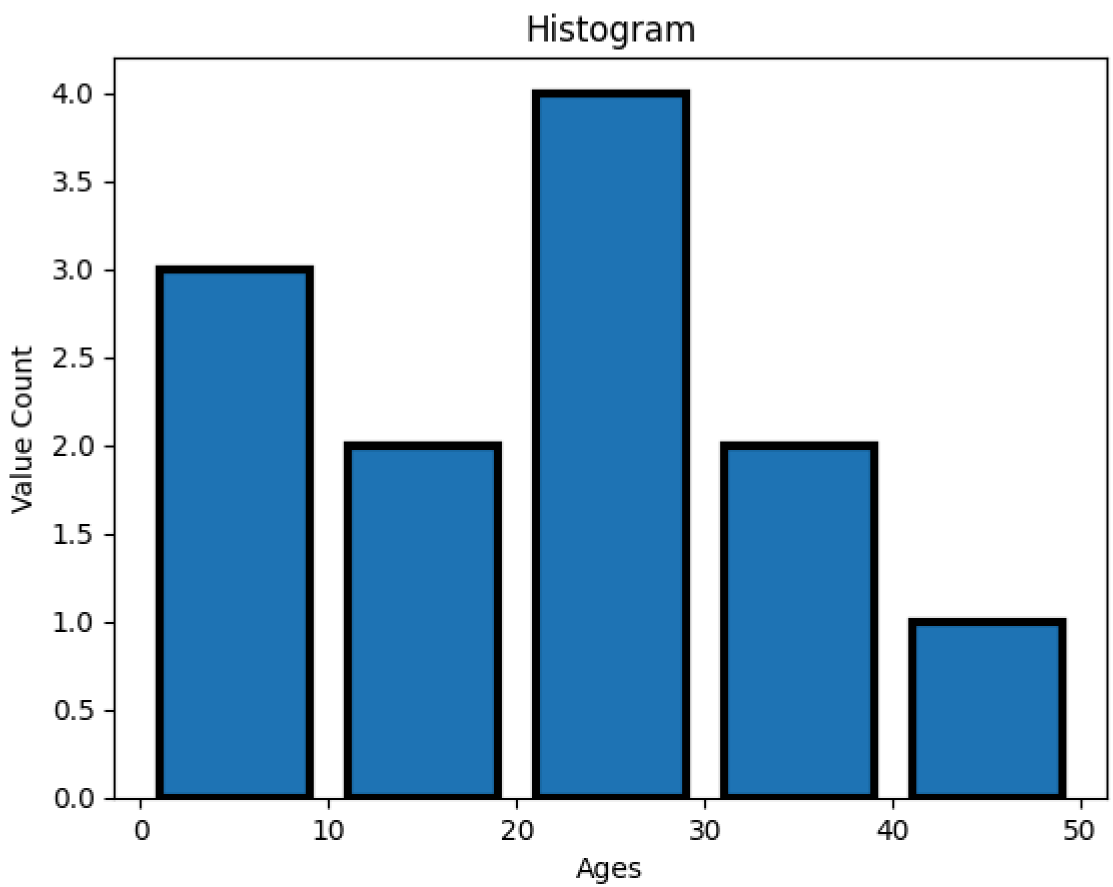




```
C:\Users\VICTUS\AppData\Local\Temp\ipykernel_23848\2296368462.py:13: UserWarning: No artists with labels found to put in legen
d.  Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
  plt.legend()
```

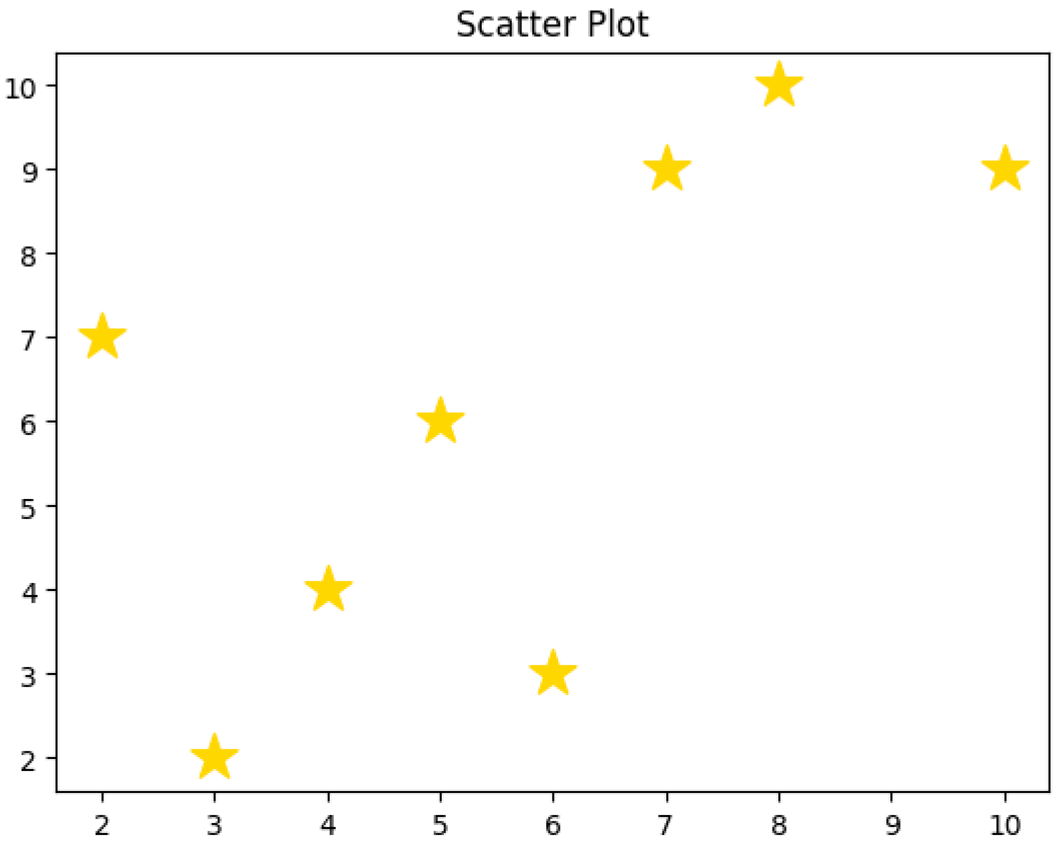


In [62]:
```
#Histogram
ages = [35,10,2,8,5,15,20,28,24,27,45,30]
bins = [0,10,20,30,40,50]

plt.hist(ages,bins,histtype='bar',rwidth=0.8,edgecolor='k',linewidth=3)
plt.title("Histogram")
plt.xlabel('Ages')
plt.ylabel('Value Count')
plt.show()
```

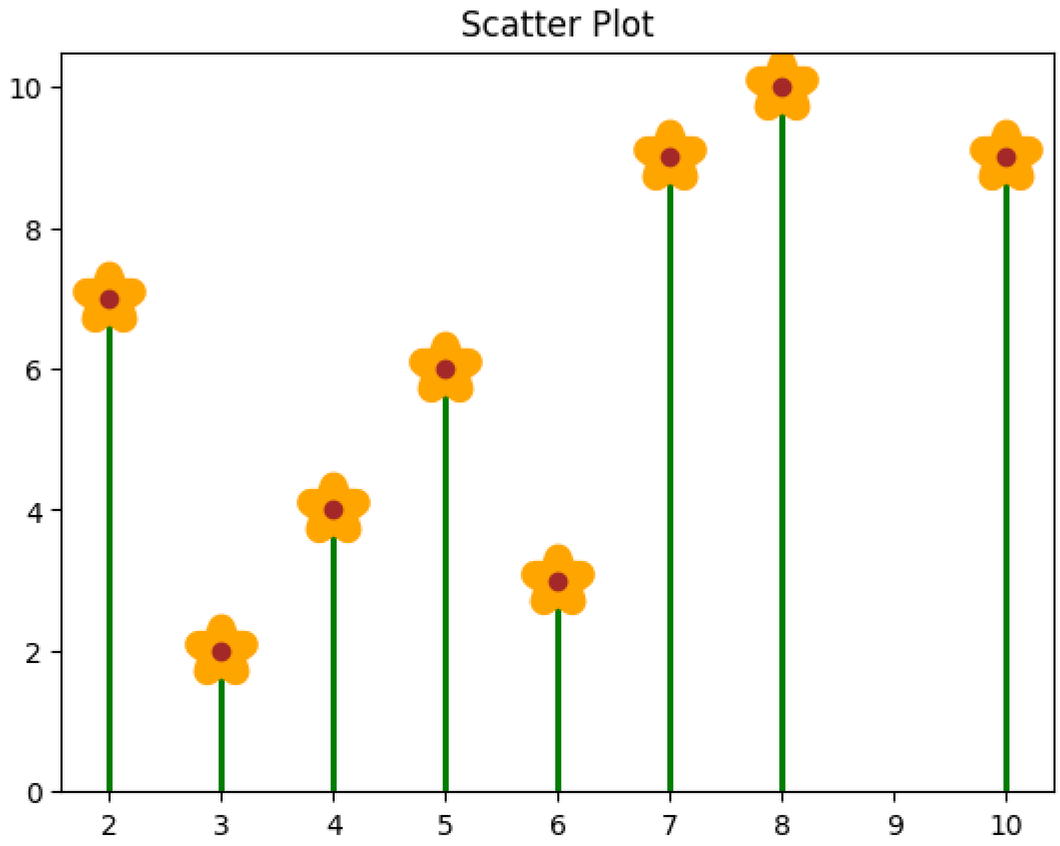


In [84]:

```
#Scatter Plot
x = [3,5,4,7,2,10,6,8]
y = [2,6,4,9,7,9,3,10]

plt.scatter(x,y,s=300,color='gold',marker='*')
plt.title("Scatter Plot")
plt.show()
```

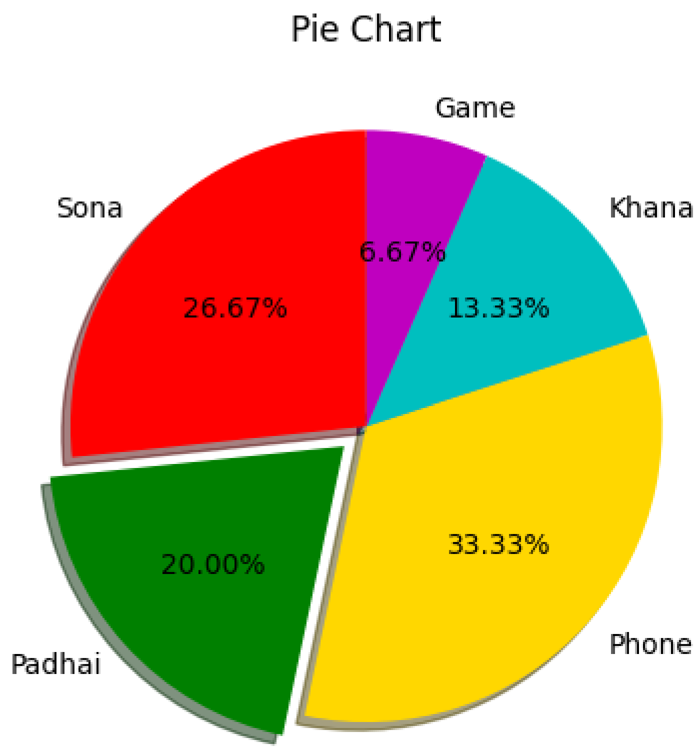


In [98]:
```
x = [3,5,4,7,2,10,6,8]
y = [2,6,4,9,7,9,3,10]

plt.bar(x,y,color='g',width=0.05)
plt.scatter(x,y,s=300,color='gold',marker='*',edgecolors='orange',linewidths=10)
plt.scatter(x,y,color='brown')

plt.title("Scatter Plot")
plt.show()
```



In [120…

```
#Pie Chart
slices = [8,6,10,4,2]
activities = ['Sona','Padhai','Phone','Khana','Game']
cols = ['r','g','gold','c','m']

plt.pie(slices,labels=activities,colors=cols,startangle=90,
        shadow=True,explode=(0,0.1,0,0,0),autopct="%1.2f%%")
plt.title("Pie Chart")
plt.show()
```


Pie Chart
Game
Sona
Khana
6.67%
26.67%
13.33%
20.00%
33.33%
Padhai
Phone

```
In [ ]:
```