Importing Libraries

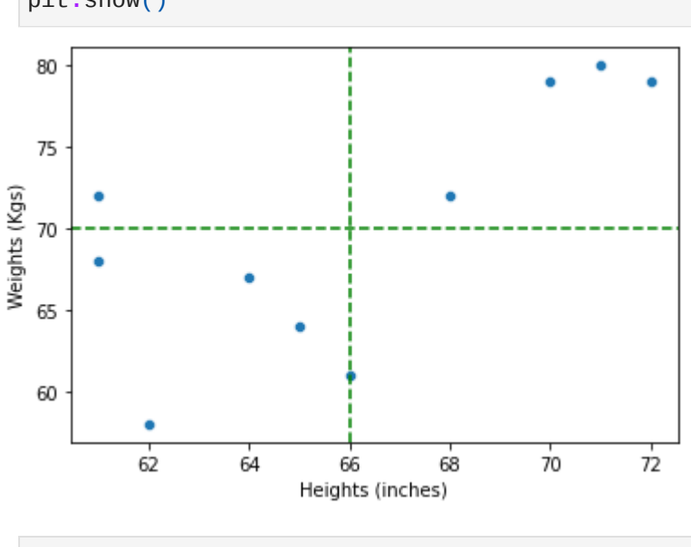
In [5]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import pearsonr, spearmanr
from scipy.stats import poisson,binom
```

Correlation

In [10]:
```python
# Ravi -->
heights = np.array([68, 62, 64, 61, 70, 66, 61, 65, 71, 72]) # inches
weights = np.array([72, 58, 67, 72, 79, 61, 68, 64, 80, 79]) # kgs

sns.scatterplot(x=heights, y=weights)
plt.axvline(heights.mean(),color="green",linestyle="--")
plt.axhline(weights.mean(),color="green",linestyle="--")

plt.xlabel("Heights (inches)")
plt.ylabel("Weights (Kgs)")
plt.show()
```



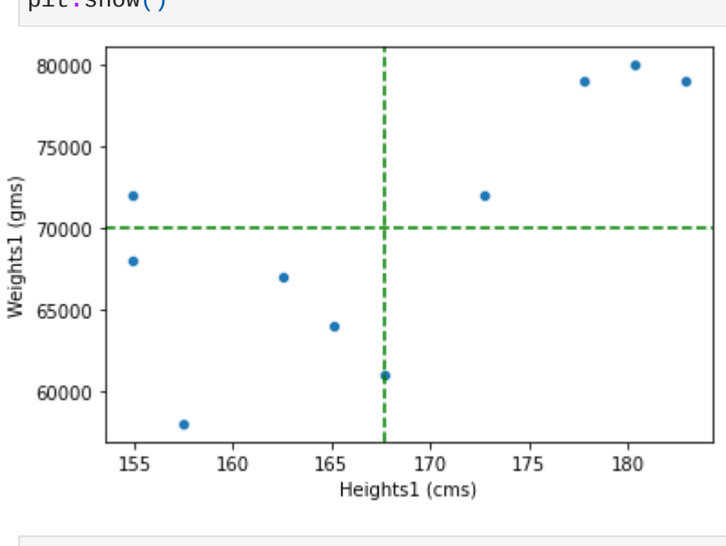
In [11]:
```python
covariance=(((heights-heights.mean())*(weights-weights.mean())).sum())/len(heights)
print("Covariance : ",covariance)
```

```
Covariance :  20.4
```

In [12]:
```python
# Rishav --> USA
heights1 = np.array([68, 62, 64, 61, 70, 66, 61, 65, 71, 72])*2.54 # cms
weights1 = np.array([72, 58, 67, 72, 79, 61, 68, 64, 80, 79])*1000 # gms

sns.scatterplot(x=heights1, y=weights1)
plt.axvline(heights1.mean(),color="green",linestyle="--")
plt.axhline(weights1.mean(),color="green",linestyle="--")

plt.xlabel("Heights1 (cms)")
plt.ylabel("Weights1 (gms)")
plt.show()
```



In [13]:
```python
covariance1=(((heights1-heights1.mean())*(weights1-weights1.mean())).sum())/len(heights1)
print("Covariance : ",covariance1)
```

```
Covariance :  51816.00000000001
```

In [14]:
```python
covariance/(heights.std()*weights.std())
```

Out[14]: `0.7094289771951879`

In [15]:
```python
covariance1/(heights1.std()*weights1.std())
```

Out[15]: `0.7094289771951878`

In [18]:
```python
np.corrcoef(weights,heights)
```

Out[18]:
```
array([[1.        , 0.70942898],
       [0.70942898, 1.        ]])
```

In [19]:
```python
np.corrcoef(weights,heights)[0,1]
```

Out[19]: `0.7094289771951878`

In [20]:
```python
pearsonr(weights,heights)
```

Out[20]: `(0.7094289771951878, 0.021575164034828447)`

In [38]:
```python
spearmanr(weights,heights)
```

Out[38]: `SpearmanrResult(correlation=0.6024493002399199, pvalue=0.06529313205986141)`

In [23]:
```python
df=pd.DataFrame([heights,weights]).transpose()
df
```

Out[23]:

	0	1
0	68	72
1	62	58
2	64	67
3	61	72
4	70	79
5	66	61
6	61	68
7	65	64
8	71	80
9	72	79

In [30]:
```python
heights_ranks=df[0].rank()
weights_ranks=df[1].rank()
```

In [31]:
```python
heights_ranks
```

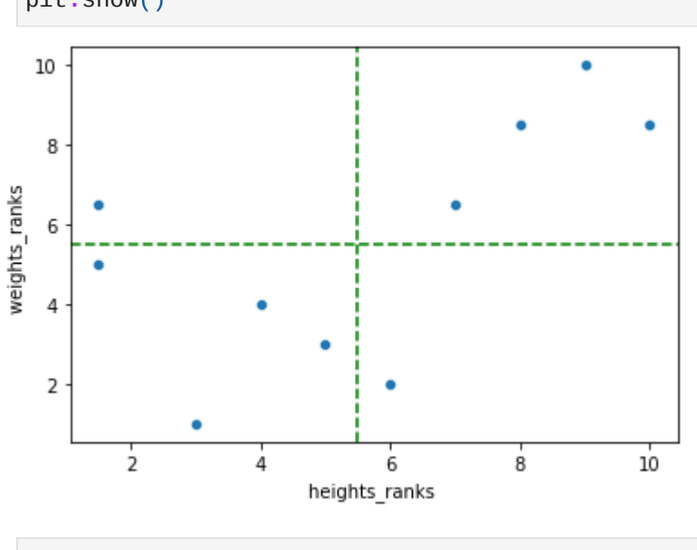
Out[31]:
```
0     7.0
1     3.0
2     4.0
3     1.5
4     8.0
5     6.0
6     1.5
7     5.0
8     9.0
9    10.0
Name: 0, dtype: float64
```

In [32]:
```python
weights_ranks
```

Out[32]:
```
0     6.5
1     1.0
2     4.0
3     6.5
4     8.5
5     2.0
6     5.0
7     3.0
8    10.0
9     8.5
Name: 1, dtype: float64
```

In [35]:
```python
sns.scatterplot(x=heights_ranks, y=weights_ranks)
plt.axvline(weights_ranks.mean(),color="green",linestyle="--")
plt.axhline(weights_ranks.mean(),color="green",linestyle="--")

plt.xlabel("heights_ranks ")
plt.ylabel("weights_ranks ")
plt.show()
```



In [37]:
```python
pearsonr(heights_ranks,weights_ranks)
```

Out[37]: `(0.6024493002399198, 0.06529313205986152)`

In [39]:
```python
df=pd.read_csv("Sachin_ODI.csv")
df
```

Out[39]:

	runs	NotOut	mins	bf	fours	sixes	sr	Inns	Opp	Ground	Date	Winner	Won	century
0	13	0	30	15	3	0	86.66	1	New Zealand	Napier	1995-02-16	New Zealand	False	False
1	37	0	75	51	3	1	72.54	2	South Africa	Hamilton	1995-02-18	South Africa	False	False
2	47	0	65	40	7	0	117.50	2	Australia	Dunedin	1995-02-22	India	True	False
3	48	0	37	30	9	1	160.00	2	Bangladesh	Sharjah	1995-04-05	India	True	False
4	4	0	13	9	1	0	44.44	2	Pakistan	Sharjah	1995-04-07	Pakistan	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
355	14	0	34	15	2	0	93.33	2	Australia	Sydney	2012-02-26	Australia	False	False
356	39	0	45	30	5	0	130.00	2	Sri Lanka	Hobart	2012-02-28	India	True	False
357	6	0	25	19	1	0	31.57	1	Sri Lanka	Dhaka	2012-03-13	India	True	False
358	114	0	205	147	12	1	77.55	1	Bangladesh	Dhaka	2012-03-16	Bangladesh	False	True
359	52	0	93	48	5	1	108.33	2	Pakistan	Dhaka	2012-03-18	India	True	False

360 rows × 14 columns

In [40]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 360 entries, 0 to 359
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   runs     360 non-null    int64
 1   NotOut   360 non-null    int64
 2   mins     360 non-null    object
 3   bf       360 non-null    int64
 4   fours    360 non-null    int64
 5   sixes    360 non-null    int64
 6   sr       360 non-null    float64
 7   Inns     360 non-null    int64
 8   Opp      360 non-null    object
 9   Ground   360 non-null    object
 10  Date     360 non-null    object
 11  Winner   360 non-null    object
 12  Won      360 non-null    bool
 13  century  360 non-null    bool
dtypes: bool(2), float64(1), int64(6), object(5)
memory usage: 34.6+ KB
```

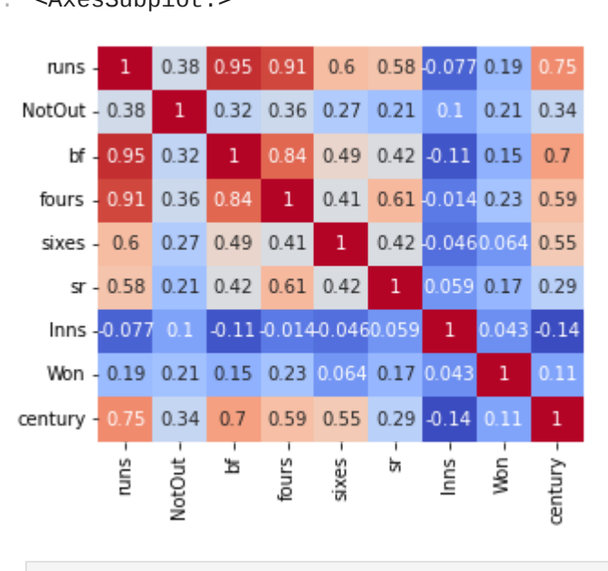
In [41]:
```python
df.corr()
```

Out[41]:

	runs	NotOut	bf	fours	sixes	sr	Inns	Won	century
runs	1.000000	0.377869	0.953558	0.907861	0.604078	0.576354	-0.076998	0.188319	0.751176
NotOut	0.377869	1.000000	0.324253	0.360667	0.266437	0.210590	0.104015	0.214461	0.336174
bf	0.953558	0.324253	1.000000	0.841240	0.487939	0.419307	-0.114803	0.153881	0.696394
fours	0.907861	0.360667	0.841240	1.000000	0.414766	0.613975	-0.014255	0.225120	0.591924
sixes	0.604078	0.266437	0.487939	0.414766	1.000000	0.415239	-0.045657	0.063837	0.547777
sr	0.576354	0.210590	0.419307	0.613975	0.415239	1.000000	0.058539	0.174703	0.292648
Inns	-0.076998	0.104015	-0.114803	-0.014255	-0.045657	0.058539	1.000000	0.043287	-0.137966
Won	0.188319	0.214461	0.153881	0.225120	0.063837	0.174703	0.043287	1.000000	0.108010
century	0.751176	0.336174	0.696394	0.591924	0.547777	0.292648	-0.137966	0.108010	1.000000

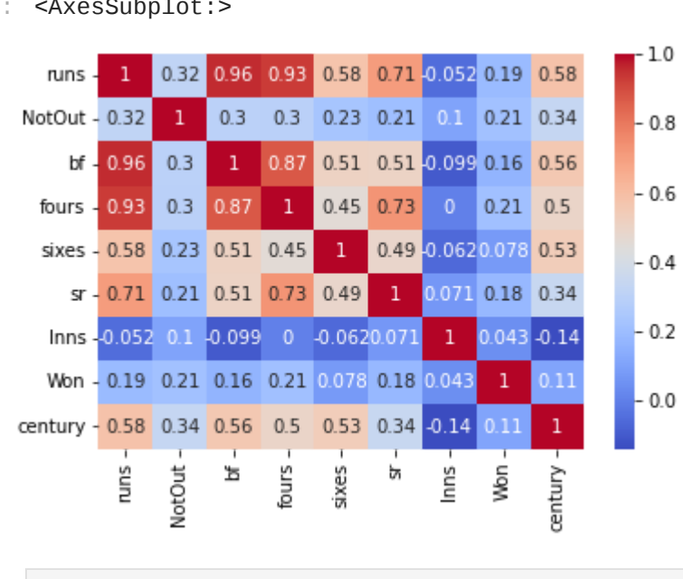
In [44]:
```python
sns.heatmap(df.corr(),annot=True,cmap="coolwarm")
```

Out[44]: `<AxesSubplot:>`



In [46]:
```python
sns.heatmap(df.corr(method="spearman"),annot=True,cmap="coolwarm")
```

Out[46]: `<AxesSubplot:>`



In [45]:
```python
df_mpg=sns.load_dataset('mpg')
df_mpg
```

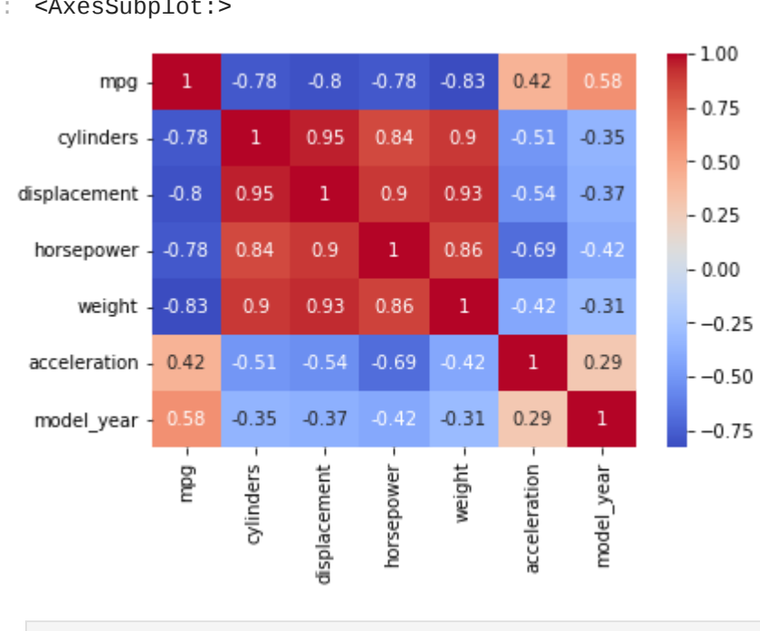
Out[45]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	usa	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86.0	2790	15.6	82	usa	ford mustang gl
394	44.0	4	97.0	52.0	2130	24.6	82	europe	vw pickup
395	32.0	4	135.0	84.0	2295	11.6	82	usa	dodge rampage
396	28.0	4	120.0	79.0	2625	18.6	82	usa	ford ranger
397	31.0	4	119.0	82.0	2720	19.4	82	usa	chevy s-10

398 rows × 9 columns

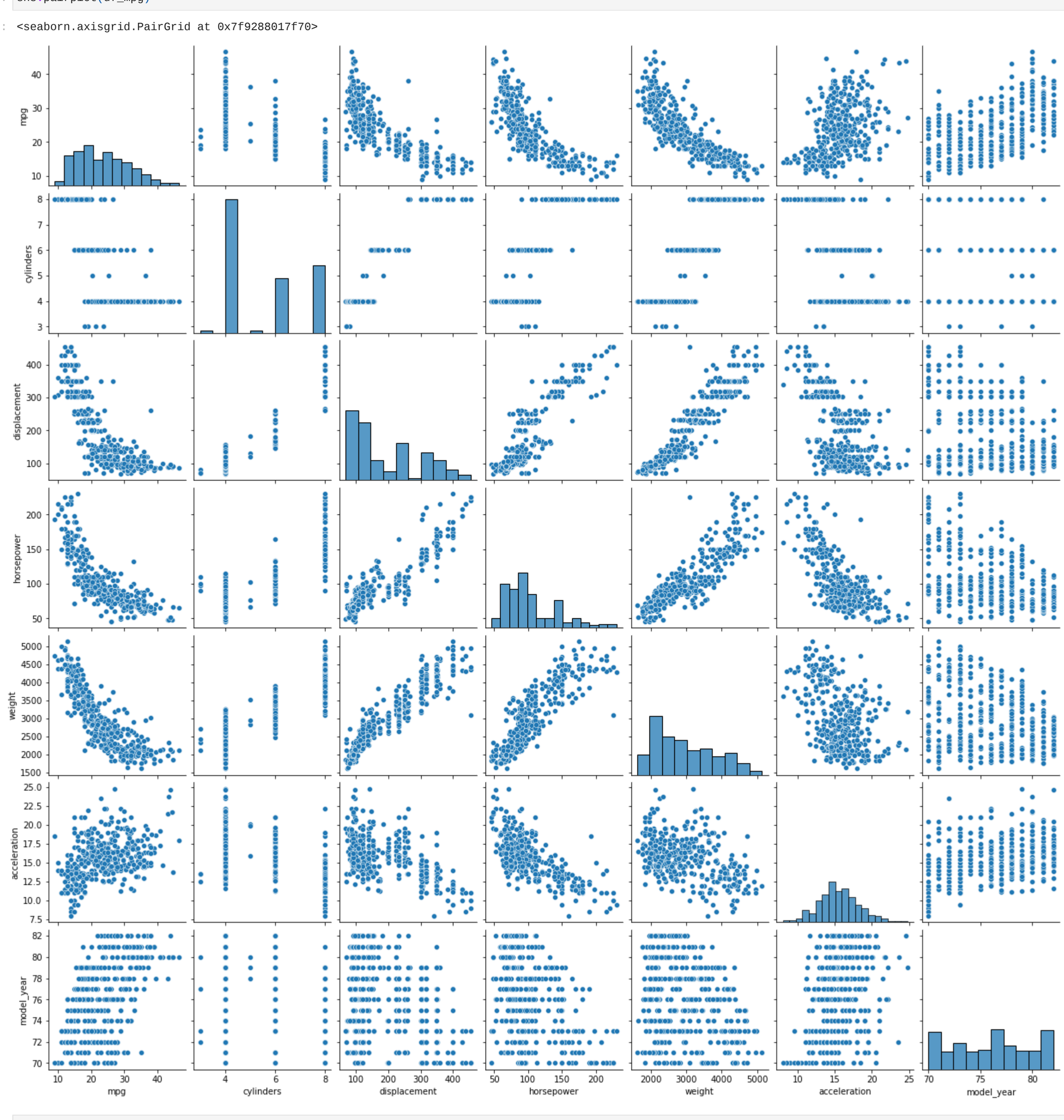
In [47]:
```python
sns.heatmap(df_mpg.corr(),annot=True,cmap="coolwarm")
```

Out[47]: `<AxesSubplot:>`



In [48]:
```python
sns.pairplot(df_mpg)
```

Out[48]: `<seaborn.axisgrid.PairGrid at 0x7f9288017f70>`



In [49]:
```python
df_mpg.isnull().sum()
```

Out[49]:
```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

In [50]:
```python
pearsonr(df_mpg["displacement"],df_mpg["mpg"])
```

Out[50]: `(-0.8042028248058989, 1.6558889101912744e-91)`

In [51]:
```python
spearmanr(df_mpg["displacement"],df_mpg["mpg"])
```

Out[51]: `SpearmanrResult(correlation=-0.8556920118178749, pvalue=2.3646353779685863e-115)`

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Poisson Distribution

In [60]:
```python
import math
def calc_poisson(k,mu):
    return ((mu**k)*((math.exp(-1*mu))))/math.factorial(k)
```

In [61]:
```python
calc_poisson(5,3)
```

Out[61]: `0.10081881344492448`

In [67]:
```python
calc_poisson(2,3)
```

Out[67]: `0.22404180765538775`

In [68]:
```python
calc_poisson(3,3)
```

Out[68]: `0.22404180765538775`

In [80]:
```python
calc_poisson(1,3)+calc_poisson(0,3)
```

Out[80]: `0.19914827347145578`

In [81]:
```python
poisson.pmf(k=1,mu=3)+poisson.pmf(k=0,mu=3)
```

Out[81]: `0.1991482734714558`

In [82]:
```python
poisson.cdf(k=1,mu=3)
```

Out[82]: `0.1991482734714558`

In [ ]:

In [62]:
```python
poisson.pmf(k=5,mu=3)
```

Out[62]: `0.10081881344492458`

In [63]:
```python
poisson.pmf(k=3,mu=3)
```

Out[63]: `0.22404180765538775`

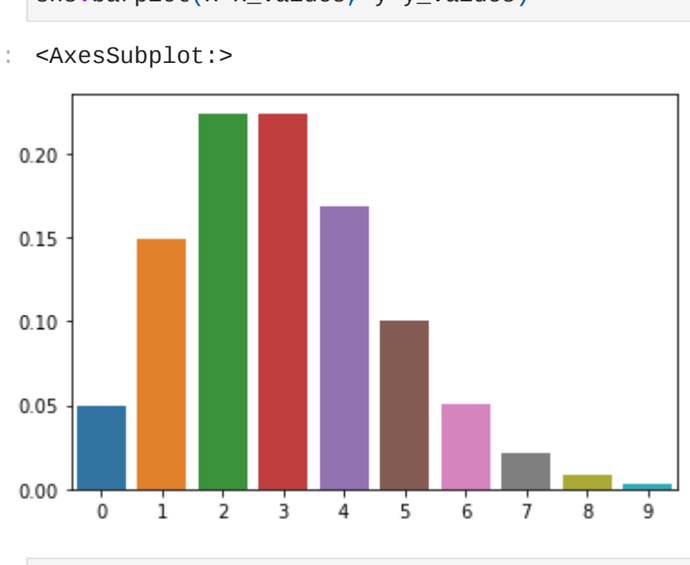
In [65]:
```python
poisson.pmf(k=2,mu=3)
```

Out[65]: `0.22404180765538775`

In [64]:
```python
mu = 3

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

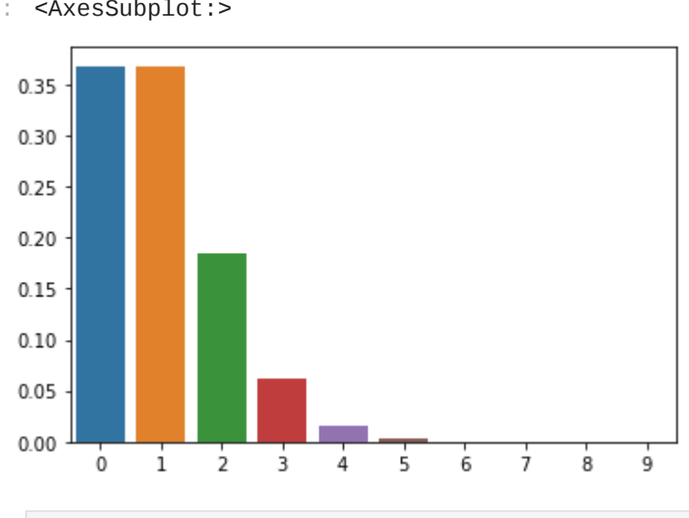
Out[64]: `<AxesSubplot:>`



In [69]:
```python
mu = 1

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

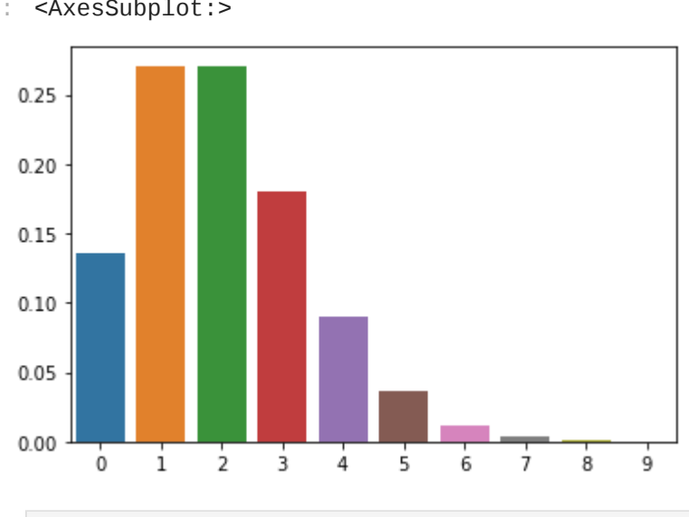
Out[69]: `<AxesSubplot:>`



In [70]:
```python
mu = 2

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

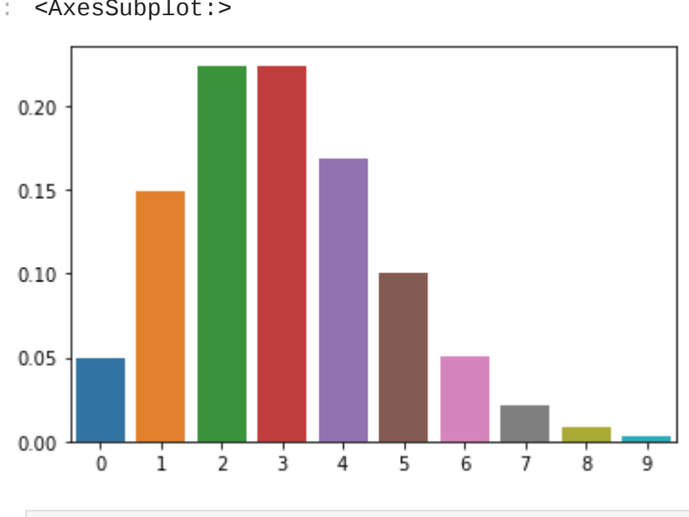
Out[70]: `<AxesSubplot:>`



In [71]:
```python
mu = 3

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

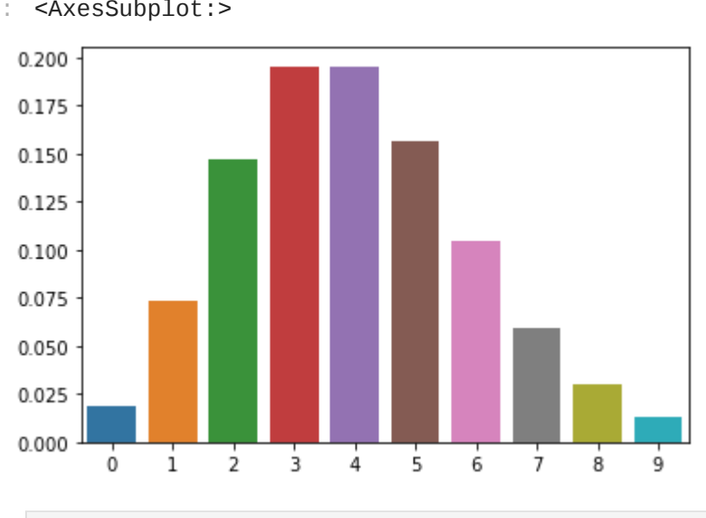
Out[71]: `<AxesSubplot:>`



In [72]:
```python
mu = 4

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

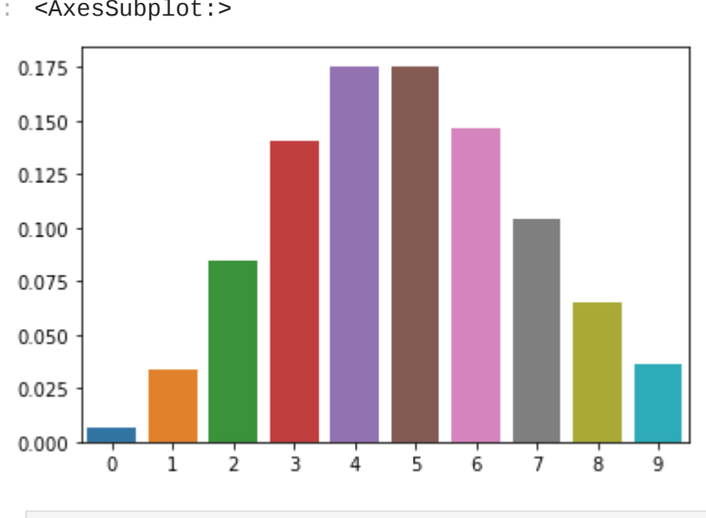
Out[72]: `<AxesSubplot:>`



In [73]:
```python
mu =5

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

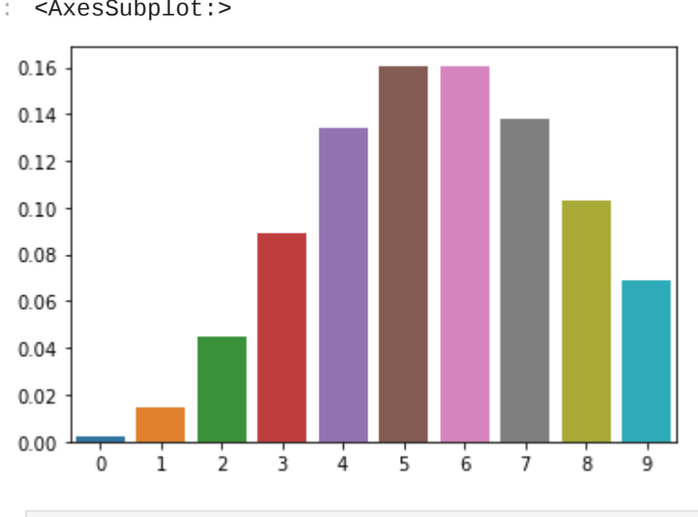
Out[73]: `<AxesSubplot:>`



In [74]:
```python
mu = 6

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

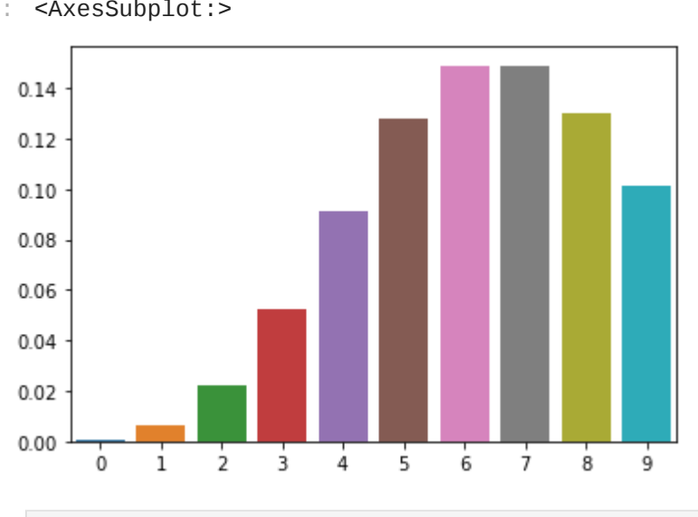
Out[74]: `<AxesSubplot:>`



In [75]:
```python
mu = 7

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

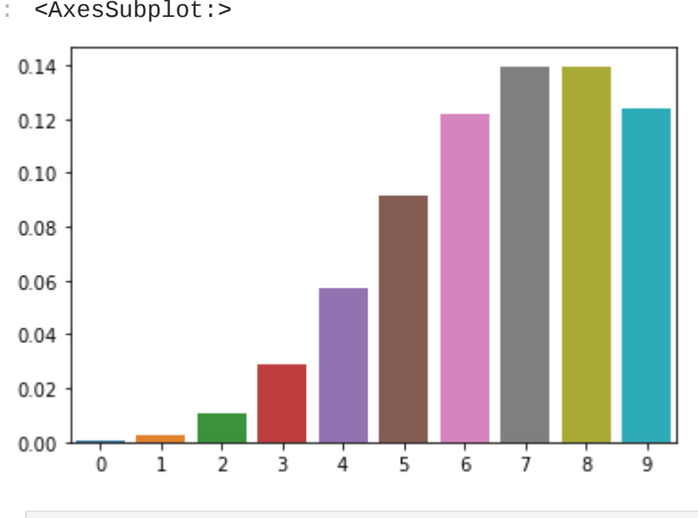
Out[75]: `<AxesSubplot:>`



In [76]:
```python
mu = 8

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

Out[76]: `<AxesSubplot:>`



In [77]:
```python
mu = 9

x_values = np.arange(0, 10)
y_values = poisson.pmf(x_values, mu=mu)
sns.barplot(x=x_values, y=y_values)
```

Out[77]: `<AxesSubplot:>`

In [ ]: