In [1]:

```
ns=6 #n(S)={1,2,3,4,5,6}
na=1 #n(A)={3}
pa=na/ns
print("probility of getting 3 is:",pa)
```

```
probility of getting 3 is: 0.16666666666666666
```

In [2]:

```
# prbility of atleast getting one head when a coin is tossed thrice
ns=(8) #n(s)={HHH,HHT,HTH,THH,TTT,TTH,THT,HTT}
na=(7) #n(A)={HHT,HHT,HTH,THH,TTH,THT,HTT}
pa=na/ns
print("probility of getting head is:",pa)
```

```
probility of getting head is: 0.875
```

In [3]:

```
# A glass jar contain 5 red ,3 blue and 2 green jelly beans.
ns=10 #n(S)={R,R,R,R,R,B,B,B,G,G,G}
na=7
pa=na/ns
print("prboility of ")
```

```
prboility of
```

In [4]:

```
p=0.7*0.5
print("prboililty that they will be alive after 20 year is:",p)
```

```
prboililty that they will be alive after 20 year is: 0.35
```

In [5]:

```
def event_probability(n,s):
    return n/s
```

In [6]:

```
# A fair die is tossed twice.find the prboility of getting
# a 4 or 5 on the first toss and a 1,2 or 3 in the second toss.
pa=event_probability(2,6)
pb=event_probability(3,6)
p=pa*pb
print("probability of getting a 4 or 5 on the first toss and a 1,2,or 3 in the second toss
```

```
probability of getting a 4 or 5 on the first toss and a 1,2,or 3 in the seco
nd toss is: 0.16666666666666666
```

In [7]:

```
def event_probability(n,s):
    return n/s
pw=event_probability(5,10)
pb=event_probability(3,9)
pg=event_probability(2,8)
p=pw*pb*pg
print("probability of obtaining white,black,and green in tha order :",p)
```

```
probability of obtaining white,black,and green in tha order : 0.041666666666
666664
```

In [8]:

```
#sample space
cards=52
heart=13
clubs=13
heart_or_club=event_probability(heart,cards)+event_probability(clubs,cards)
print(heart_or_club)
```

```
0.5
```

In [9]:

```
ace=4
king=4
queen=4
ace_king_or_queen=event_probability(ace,cards)+event_probability(king,cards)+event_probabil
print(ace_king_or_queen)
```

```
0.23076923076923078
```

In [10]:

```
cards=52
ace=4
heart=13
ace_of_hearts=1
heart_or_ace=event_probability(heart,cards)+event_probability(ace,cards)-event_probability(
print(heart_or_ace)
```

```
0.3076923076923077
```

In [11]:

```
cards=52
ace=4
heart=13
ace_of_hearts=1
event_probability(heart,cards)+event_probability(ace,cards)-event_probability(ace_of_hearts

print(round(heart_or_ace,3))
```

```
0.308
```

In [12]:

```
red=26
facecard=12
red_facecard=6
r=event_probability(red,cards)+event_probability(facecard,cards)-event_probability(red_face
print(round(r,2))
```

```
0.62
```

In [13]:

```
ns=6
na=1
pa=1-na/ns
print("probility of getting 5 is:",pa)
```

```
probility of getting 5 is: 0.8333333333333334
```

In [14]:

```
card=52
j=4
ace=4
pj=event_probability(j,52)
pa=event_probability(ace,51)
pa_given_j=(pj*pa)/pj
print(pa_given_j)
```

```
0.0784313725490196
```

In [15]:

```
import pandas as pd
import numpy as np #conditional probability
df=pd.read_csv("C:/Users/MSCIT/Downloads/student-mat - student-mat (1).csv")
df.head(3)
```

Out[15]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	

3 rows × 33 columns

In [16]:

```
len(df)
```

Out[16]:

```
395
```

In [17]:

```
df['grade_A']=np.where(df['G3']*5>=80,1,0)
df
```

Out[17]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	freetime
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	3
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	3
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	3
3	GP	F	15	U	GT3	T	4	2	health	services	...	2
4	GP	F	16	U	GT3	T	3	3	other	other	...	3
...	...	...	...	...	...	...	...	...	...	...	...	...
390	MS	M	20	U	LE3	A	2	2	services	services	...	5
391	MS	M	17	U	LE3	T	3	1	services	services	...	4
392	MS	M	21	R	GT3	T	1	1	other	other	...	5
393	MS	M	18	R	LE3	T	3	2	services	other	...	4
394	MS	M	19	U	LE3	T	1	1	other	at_home	...	2

395 rows × 34 columns

In [18]:

```
df['high_absences']=np.where(df['absences']>=10,1,0)
df
```

Out[18]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	goout
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	3
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	2
3	GP	F	15	U	GT3	T	4	2	health	services	...	2
4	GP	F	16	U	GT3	T	3	3	other	other	...	2
...	...	...	...	...	...	...	...	...	...	...	...	...
390	MS	M	20	U	LE3	A	2	2	services	services	...	4
391	MS	M	17	U	LE3	T	3	1	services	services	...	5
392	MS	M	21	R	GT3	T	1	1	other	other	...	3
393	MS	M	18	R	LE3	T	3	2	services	other	...	1
394	MS	M	19	U	LE3	T	1	1	other	at_home	...	3

395 rows × 35 columns

In [19]:

```
df['count']=1
df
```

Out[19]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	Dalc	W
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	1	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	1	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	2	
3	GP	F	15	U	GT3	T	4	2	health	services	...	1	
4	GP	F	16	U	GT3	T	3	3	other	other	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
390	MS	M	20	U	LE3	A	2	2	services	services	...	4	
391	MS	M	17	U	LE3	T	3	1	services	services	...	3	
392	MS	M	21	R	GT3	T	1	1	other	other	...	3	
393	MS	M	18	R	LE3	T	3	2	services	other	...	3	
394	MS	M	19	U	LE3	T	1	1	other	at_home	...	3	

395 rows × 36 columns

In [20]:

```
df=df[['grade_A','high_absences','count']]
df.head()
```

Out[20]:

	grade_A	high_absences	count
0	0	0	1
1	0	0	1
2	0	1	1
3	0	0	1
4	0	0	1

In [21]:

```
df=df[['grade_A','high_absences','count']]
df.head(395)
```

Out[21]:

	grade_A	high_absences	count
0	0	0	1
1	0	0	1
2	0	1	1
3	0	0	1
4	0	0	1
...	...	...	...
390	0	1	1
391	1	0	1
392	0	0	1
393	0	0	1
394	0	0	1

395 rows × 3 columns

In [22]:

```
final=pd.pivot_table(
     df,
     values='count',
     index=['grade_A'],
     columns=['high_absences'],
     aggfunc=np.size,
     fill_value=0
)
print(final)
```

```
high_absences    0   1
grade_A
0              277  78
1               35   5
```

In [27]:

```
total=final.iloc[0,0]+final.iloc[0,1]+final.iloc[1,0]+final.iloc[1,1]
p_a=(final.iloc[1,0]+final.iloc[1,1])/total
p_a
```

Out[27]:

```
0.10126582278481013
```

In [34]:

```
total=final.iloc[0,0]+final.iloc[0,1]+final.iloc[1,0]+final.iloc[1,1]
p_b=(final.iloc[0,1]+final.iloc[1,1])/total
p_b
```

Out[34]:

```
0.21012658227848102
```

In [36]:

```
p_c=(final.iloc[1,1])/total
p_c
```

Out[36]:

```
0.012658227848101266
```

In [37]:

```
p_d=p_c/p_b
p_d
```

Out[37]:

```
0.060240963855421686
```

In [ ]:

```

```