In [13]:

```
#probability of getting 3 when a die is rolled
ns=6#n(s) = {1,2,3,4,5,6}
na=1#n(A)={3}
pa=na/ns#P(A)
print("probability of getting 3 when a die is rolled is:",pa)
```

```
probability of getting 3 when a die is rolled is: 0.16666666666666666
```

In [31]:

```
#probability of getting atlear one head when a coin is tossed thrice.
ns=8 #n(s)={HHH,HHT,HTT,TTT,HTH,THH,THT,TTH}
na = 7 #n(a)={HHT,HTT,TTT,HTH,THH,THT,TTH}
pa = na/ns
pa

```

```
0.875
```

In [32]:

```
#if the probability that person A will alive in 20 year
#is 0.7 and the probability that person B will be alive in 20 year is 0.5
#what is the probability that they will both br alive in 20 year?
p=0.75*0.5
print("probability that they both alive in 20 year:",p)
```

```
probability that they both alive in 20 year: 0.375
```

In [35]:

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

In [36]:

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

```
probability of getting 4 or 5 in the first toss and 1,2 or 3 om the second toss is: 0.16666666666666666
```

In [37]:

```
# A bag contain 5 white marble, 3 black marble and 2 green marbles.
# In each draw, a marble is drawn from the bag and not replaced.
#In three draws, find the probability of obtaining white,black and green in order.
pa = event_probability(5,10)
pb = event_probability(3,9)
pc = event_probability(2,8)
p = pa*pb*pc
print("probability of getting white, black, green in order :",p)
```

```
probability of getting white, black, green in order : 0.041666666666666664
```

In [39]:

```
#sample space
cards = 52
#calculate the probability of drawing heart of club
hearts = 13
clubs=13
heart_or_club=event_probability(hearts,cards)+event_probability(clubs,cards)
print(heart_or_club)
```

```
0.5
```

In [40]:

```
#calculate the probability of drawing an ace, queen or ace
cards=52
ace=4
queen=4
king=4
ace_queen_or_king = event_probability(ace,cards)+event_probability(queen,cards)+ev
print("probability of drawing a ace,queen or king:",ace_queen_or_king)
```

```
probability of drawing a ace,queen or king: 0.23076923076923078
```

In [43]:

```
#calculate the probability of drawing a heart or an ace
cards=52
hearts=13
ace=4
ace_of_hearts=1
hearts_or_aces=event_probability(hearts,cards)+event_probability(ace,cards)-event_
print("probability of getting heart or an ace:",round(hearts_or_aces,2))
```

```
probability of getting heart or an ace: 0.31
```

In [44]:

```
#calculate the probability of drawing red cards or face cards
cards=52
red=26
face_cards=12
red_face_cards=6
red_cards_or_face_cards=event_probability(red,cards)+event_probability(face_cards,
print("probability of drawing red cards or face cards:",round(red_cards_or_face_ca
```

```
probability of drawing red cards or face cards: 0.62
```

In [45]:

```
#probability of getting a fair die is rolled
ns=6
na=1
pa=na/ns
print("probability of not getting a fair die is rolled is:",round(1-pa,2))
```

```
probability of not getting a fair die is rolled is: 0.83
```

In [46]:

```
#suppose you are draw 2 cards from deck.
#you win if you get ace given that you draw a jeck in first draw.
cards=52
jacks=4
aces=4
pj=event_probability(jacks,52)
pa=event_probability(aces,51)
pa_given_jack=(pj*pa)/pj
print("probability of winning is:",round(pa_given_jack,4))
```

```
probability of winning is: 0.0784
```

Conditional Probability

In [3]:

```
import numpy as np
import pandas as pd
df = pd.read_csv("student-mat.csv")
df.head(3)
```

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3

```
3 rows × 33 columns
```

In [4]:

```
#calculate a probability student get 80+ in math  given that he misses 10 or more
df["grade_A"]=np.where(df["G3"]*5 >=80,1,0)
```

In [5]:

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

In [6]:

```
df["count"]=1
```

In [7]:

```
df = df[["grade_A","high_absenses","count"]]
df.head()
```

```
   grade_A  high_absenses  count
0  0        0              1
1  0        0              1
2  0        1              1
3  0        0              1
4  0        0              1
```

In [11]:

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


```

In [12]:

```
final
```

high_absenses	0	1
grade_A		
0	277	78
1	35	5

In [24]:

```
# A is the event getting A grade
# B is the event student absent 10 or more classes
pb=(final.iloc[0,1]+final.iloc[1,1])/(final.iloc[0,0]+final.iloc[0,1]+final.iloc[1
print(pb)
a_and_b = (final.iloc[1,1])/(final.iloc[0,0]+final.iloc[0,1]+final.iloc[1,0]+final
print(a_and_b)
```

```
0.21012658227848102
0.012658227848101266
```

In [26]:

```
A_given_B = (a_and_b)/(pb)
A_given_B
```

```
0.060240963855421686
```

In [29]:

```
print("The probability of student getting at leat A grade(80%+) final grade,given
      more class is:",round(A_given_B,4))
```

```
The probability of student getting at leat A grade(80%+) final grade,given missing 10 or more class is: 0.0602
```

In [ ]:

```

```