Assignment 05 for MT205 in Fall 2020

In [1]:
```
j = {}
```

In [2]:
```
total_count = 0

for d1 in range(1, 7):
    for d2 in range(1, 7):
        # print((d1, d2))
        try:
            X = max(d1, d2)    # Actual Random Variables
            Y = d1 + d2
            j[(X, Y)] += 1
        except KeyError:
            j[(X, Y)]  = 1
        total_count += 1
```

In [3]:
```
j
```

Out[3]:
```
{(1, 2): 1,
 (2, 3): 2,
 (3, 4): 2,
 (4, 5): 2,
 (5, 6): 2,
 (6, 7): 2,
 (2, 4): 1,
 (3, 5): 2,
 (4, 6): 2,
 (5, 7): 2,
 (6, 8): 2,
 (3, 6): 1,
 (4, 7): 2,
 (5, 8): 2,
 (6, 9): 2,
 (4, 8): 1,
 (5, 9): 2,
 (6, 10): 2,
 (5, 10): 1,
 (6, 11): 2,
 (6, 12): 1}
```

In [4]:
```
x_vals = list(range(1, 7))
y_vals = list(range(1, 13))

print("X: ", x_vals)
print("Y: ", y_vals)
```

```
X:  [1, 2, 3, 4, 5, 6]
Y:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

In [5]:
```
# Create the joint probability table from the frequencies -- show it and also normalize it

print (" - Y -  ", end = '')
for y in y_vals:
        print(y, end = "\t")
print (" ")

total = 0  # sanity check
print("X \n")


for x in x_vals:
    print(x, "   - ", end="\t")
    for y in y_vals:
        try:
            val = j[(x, y)] / total_count
            j[(x, y)] = val    # save the normalized value back in our table
            total += val
            print("{:.2f}".format(val), end="\t")
        except KeyError:
            print('', end='\t')



    print(" \n")

print("(Sanity check: Total: ", total)
```

```
 - Y -  1       2       3       4       5       6       7       8       9       10      11      12
X

1    -          0.03

2    -                  0.06    0.03

3    -                          0.06    0.06    0.03

4    -                                  0.06    0.06    0.06    0.03

5    -                                          0.06    0.06    0.06    0.06    0.03

6    -                                                  0.06    0.06    0.06    0.06    0.06    0.03

(Sanity check: Total:  1.0000000000000002
```

In [6]:
```
x_dist = []
for x in x_vals:
    x_dist_val = 0
    for y in y_vals:
        try:
            x_dist_val += j[(x, y)]
        except KeyError:
            pass # no worries if no value

    x_dist.append(x_dist_val)
```

In [7]:
```
for x in range(1, 7):
    print("{:.2f}".format(x), end="\t")
print(' ')
for x in x_dist:
    print("{:.2f}".format(x), end="\t")

print()
print(sum(x_dist))
```

```
1.00    2.00    3.00    4.00    5.00    6.00
0.03    0.08    0.14    0.19    0.25    0.31
1.0
```

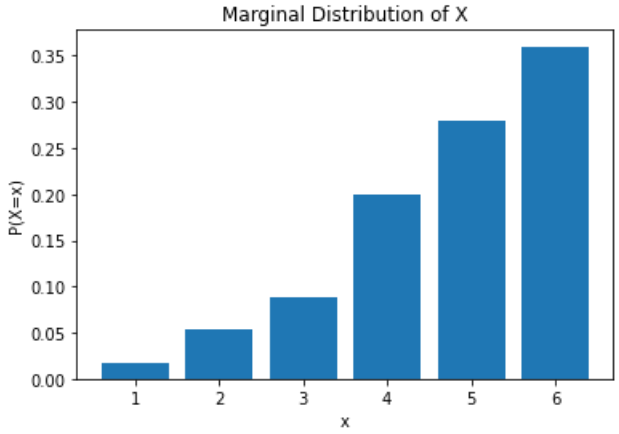
In [8]:
```
y_dist = []
for y in y_vals:
    y_dist_val = 0
    for x in x_vals:
        try:
            y_dist_val += j[(x, y)]
        except KeyError:
            pass # no worries if no value

    y_dist.append(y_dist_val)
```

In [9]:
```
for y in range(1, 13):
    print("{:.2f}".format(y), end="\t")
print(' ')
for y in y_dist:
    print("{:.2f}".format(y), end="\t")

print()
print(sum(y_dist))
```

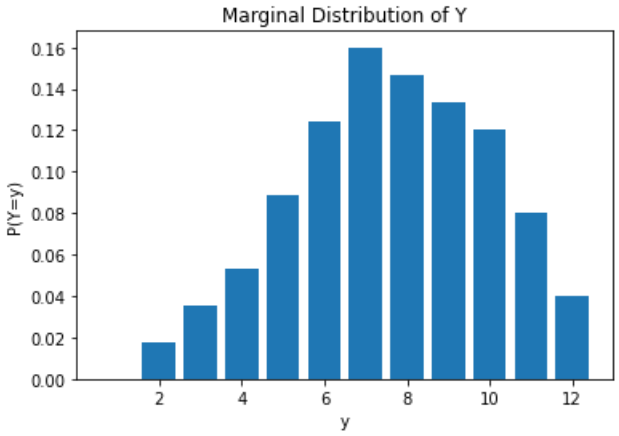
```
1.00    2.00    3.00    4.00    5.00    6.00    7.00    8.00    9.00    10.00   11.00   12.00
0.00    0.03    0.06    0.08    0.11    0.14    0.17    0.14    0.11    0.08    0.06    0.03
1.0000000000000002
```

In [10]:
```
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
```

In [30]:
```
fig = plt.figure()
rv_vals = list(range(1, max(x_vals) + 1))
plt.xlabel("x")
plt.ylabel("P(X=x)")
plt.title("Marginal Distribution of X")
plt.bar(rv_vals, x_dist)
plt.show()
```



In [31]:
```
fig = plt.figure()
rv_vals = list(range(1, max(y_vals)+1))
plt.xlabel("y")
plt.ylabel("P(Y=y)")
plt.title("Marginal Distribution of Y")
plt.bar(rv_vals, y_dist)
plt.show()
```



In [ ]:
```

```

Challenge question

In [13]:
```
j = {}
```

In [14]:
```
def prob_of(n):
    if n <= 3:
        return 0.4
    else:
        return 0.6
```

In [15]:
```
def probs_of_two(n1, n2):
    return prob_of(n1) * prob_of(n2)
```

In [16]:
```
probs_of_two(1, 5)
```

Out[16]:
```
0.24
```

In [17]:
```
j = {}
total_count = 0

for d1 in range(1, 7):
    for d2 in range(1, 7):
        # print((d1, d2))
        this_val = probs_of_two(d1, d2)
        try:
            X = max(d1, d2)
            Y = d1 + d2
            j[(X, Y)] += this_val
        except KeyError:
            j[(X, Y)]  = this_val
        total_count += this_val
```

In [18]:
```
x_vals = list(range(1, 7))
y_vals = list(range(1, 13))

print("X: ", x_vals)
print("Y: ", y_vals)
```

```
X:  [1, 2, 3, 4, 5, 6]
Y:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

In [19]:
```
print ("  - Y -   ", end = '')
for y in y_vals:
        print(y, end = "\t")
print (" ")

total = 0  # sanity check
print("X \n")


for x in x_vals:
    print(x, "    - ", end="\t")
    for y in y_vals:
        try:
            val = j[(x, y)] / total_count
            j[(x, y)] = val
            total += val
            print("{:.2f}".format(val), end="\t")
        except KeyError:
            print('', end='\t')



    print(" \n")

print("Sanity check Total: ", total)
```

```
 - Y -  1	2	3	4	5	6	7	8	9	10	11	12	
X

1    -	0.02	

2    -		0.04	0.02	

3    -			0.04	0.04	0.02	

4    -				0.05	0.05	0.05	0.04	

5    -					0.05	0.05	0.05	0.08	0.04	

6    -						0.05	0.05	0.05	0.08	0.08	0.04	

Sanity check Total:  0.9999999999999996
```

In [20]:
```
x_dist = []
for x in x_vals:
    x_dist_val = 0
    for y in y_vals:
        try:
            x_dist_val += j[(x, y)]
        except KeyError:
            pass # no worries if no value

    x_dist.append(x_dist_val)
```

In [21]:
```
for x in range(1, 7):
    print("{:.2f}".format(x), end="\t")
print(' ')
for x in x_dist:
    print("{:.2f}".format(x), end="\t")

print()
print(sum(x_dist))
```

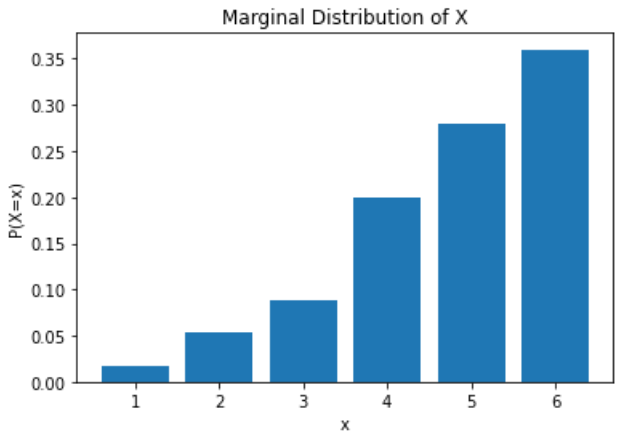
```
1.00    2.00    3.00    4.00    5.00    6.00
0.02    0.05    0.09    0.20    0.28    0.36
0.9999999999999996
```

In [22]:
```
y_dist = []
for y in y_vals:
    y_dist_val = 0
    for x in x_vals:
        try:
            y_dist_val += j[(x, y)]
        except KeyError:
            pass # no worries if no value

    y_dist.append(y_dist_val)
```

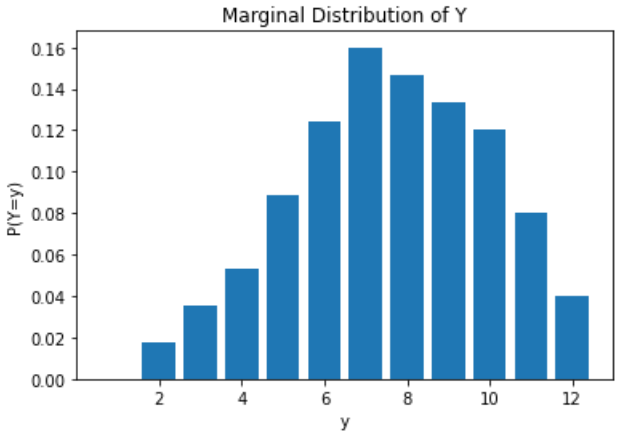
In [23]:
```
for y in range(1, 13):
    print("{:.2f}".format(y), end="\t")
print(' ')
for y in y_dist:
    print("{:.2f}".format(y), end="\t")

print()
print(sum(y_dist))
```

```
1.00    2.00    3.00    4.00    5.00    6.00    7.00    8.00    9.00    10.00   11.00   12.00
0.00    0.02    0.04    0.05    0.09    0.12    0.16    0.15    0.13    0.12    0.08    0.04
0.9999999999999996
```

In [32]:
```
fig = plt.figure()
rv_vals = list(range(1, max(x_vals) + 1))
plt.xlabel("x")
plt.ylabel("P(X=x)")
plt.title("Marginal Distribution of X")
plt.bar(rv_vals, x_dist)
plt.show()
```



In [33]:
```
fig = plt.figure()
rv_vals = list(range(1, max(y_vals)+1))
plt.xlabel("y")
plt.ylabel("P(Y=y)")
plt.title("Marginal Distribution of Y")
plt.bar(rv_vals, y_dist)
plt.show()
```



In [ ]:

In [ ]: