

Name/ID #: Aaron Garcia/030556771

Name of Assignment: Computer Assignment 1

Due Date: Sept 10, 2022

Submission Date: Aug 29, 2022

Outline:

This python code is used to simulate the 3 firms picking 3 months out of the calendar year at random. This will explain what each variable means and what it is doing to contribute to the simulation.

Sum: This aggregator variable is used by the system to keep track of the number of favorable outcomes that the program produces (When each firm picks a different month to conduct the auditing.)

K is the number of times the simulation is run, which is facilitated by “for k in range(K)”

months is the array where the 3 randomly picked months are stored

month_Number is the random month variable, where a number 1-12 is picked and stored, each number representing the corresponding months in a year.

The functionality is commented in the program itself- refer to the python file for more information.

K=10

Output=0.6

```
15 import random # Using Python's Random Function
16
17 Sum = 0 # Initialize count of number of favorable outcomes. (accumulator variable).
18
19 months = [0, 0, 0] # Identifies which month is chosen.
20
21 K = 10 # The number of experiments.
22
23 for k in range(K): # Total number of experiments to be performed.
24     # -----
25     # The experiment starts.
26     for i in range(3): # For the number of
27         month_Number = random.randrange(1,13) # Pick a random number between 1-12
28         months[i] = month_Number # Place each number at the index i
29     # The experiment ends.
30     # -----
31     # Test for favorable outcomes.
32     if (i == 2) & (months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2]):
33         Sum = Sum + 1 # Increment the count of favorable outcomes.
34     # -----
35     # The ratio of favorable outcomes to the total number of outcomes.
36     print(Sum / K)
```

Python Console

```
> Can = [list: 3] [1, 9, 3]
> Can_Number = [int] 3
> K = [int] 10
> Sum = [int] 6
> i = [int] 2
> k = [int] 9
> month_Number = [int] 1
> months = [list: 3] [9, 4, 1]
> Special Variables
```

K=50

Output=.82

```
15 import random_# Using Python's Random Function
16
17 Sum = 0_# Initialize count of number of favorable outcomes. (accumulator variable).
18
19 months = [0, 0, 0]_# Identifies which month is chosen.
20
21 K = 50_# The number of experiments.
22
23 for k in range(K):_# Total number of experiments to be performed.
24     # -----
25     # The experiment starts.
26     for i in range(3):_# For the number of
27         month_Number = random.randrange(1,13)_# Pick a random number between 1-12
28         months[i] = month_Number_# Place each number at in the list
29     # The experiment ends.
30     # -----
31     # Test for favorable outcomes.
32     if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
33         Sum = Sum + 1_# Increment the count of favorable outcomes.
34     # -----
35     # The ratio of favorable outcomes to the total number of outcomes.
36     print(Sum / K)
```

Python Console

```
> Can = (list: 3) [1, 9, 3]
> Can_Number = (int) 3
> K = (int) 50
> Sum = (int) 41
> i = (int) 2
> k = (int) 49
> month_Number = (int) 6
> months = (list: 3) [2, 1, 6]
> Special Variables
```

K=100

Output=.72

```
import random_# Using Python's Random Function

Sum = 0_# Initialize count of number of favorable outcomes. (accumulator variable).

months = [0, 0, 0]_# Identifies which month is chosen.

K = 100_# The number of experiments.

for k in range(K):_# Total number of experiments to be performed.
    # -----
    # The experiment starts.
    for i in range(3):
        month_Number = random.randrange(1,13)
        months[i] = month_Number
    # The experiment ends.
    # -----
    # Test for favorable outcomes.
    if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
        Sum = Sum + 1_# Increment the count of favorable outcomes.
    # -----
    # The ratio of favorable outcomes to the total number of outcomes.
    print(Sum / K)
```

Python Console

```
# The experiment starts.
for i in range(3):
    month_Number = random.randrange(1,13)
    months[i] = month_Number
# The experiment ends.
# -----
# Test for favorable outcomes.
if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
    Sum = Sum + 1_# Increment the count of favorable outcomes.
# -----
# The ratio of favorable outcomes to the total number of outcomes.
print(Sum / K)
```

Can = (list: 3) [1, 9, 3]
Can_Number = (int) 3
K = (int) 100
Sum = (int) 72
i = (int) 2
k = (int) 99
month_Number = (int) 11
months = (list: 3) [12, 8, 11]
Special Variables

K=500
Output=.81

```
import random_# Using Python's Random Function

Sum = 0_# Initialize count of number of favorable outcomes. (accumulator variable).

months = [0, 0, 0]_# Identifies which month is chosen.

K = 500_# The number of experiments.

for k in range(K):_# Total number of experiments to be performed.
    # -----
    # The experiment starts.
    for i in range(3):
        month_Number = random.randrange(1,13)
        months[i] = month_Number
    # The experiment ends.
    # -----
    # Test for favorable outcomes.
    if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
        Sum = Sum + 1_# Increment the count of favorable outcomes.
    # -----
    # The ratio of favorable outcomes to the total number of outcomes.
    print(Sum / K)
```

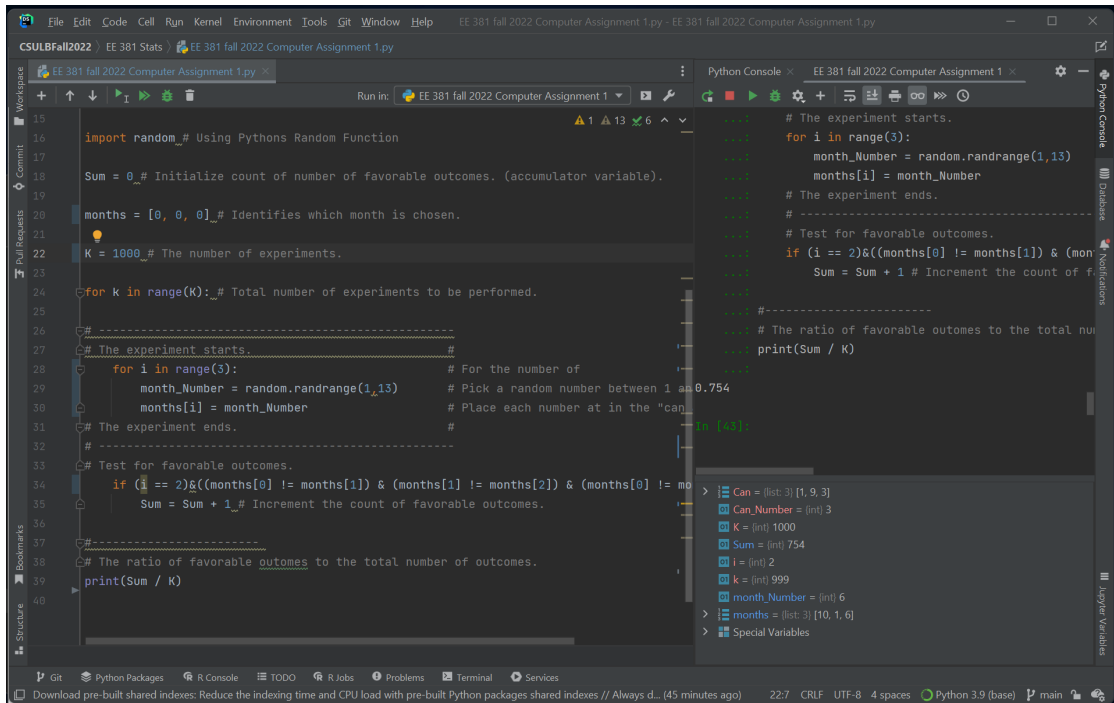
Python Console

```
# The experiment starts.
for i in range(3):
    month_Number = random.randrange(1,13)
    months[i] = month_Number
# The experiment ends.
# -----
# Test for favorable outcomes.
if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
    Sum = Sum + 1_# Increment the count of favorable outcomes.
# -----
# The ratio of favorable outcomes to the total number of outcomes.
print(Sum / K)
```

Can = (list: 3) [1, 9, 3]
Can_Number = (int) 3
K = (int) 500
Sum = (int) 405
i = (int) 2
k = (int) 499
month_Number = (int) 1
months = (list: 3) [2, 7, 1]
Special Variables

K=1000

Output=.754



The screenshot displays a Jupyter Notebook environment with a dark theme. The main area shows a Python script for a simulation. The script imports the random module and initializes variables: Sum = 0, months = [0, 0, 0], and K = 1000. It then enters a loop for k in range(K). Inside this loop, it starts an experiment by choosing three random months (0, 1, or 2) and stores them in the months list. It then tests for favorable outcomes based on specific conditions involving the months. If the conditions are met, it increments Sum. Finally, it prints the ratio of favorable outcomes to the total number of experiments (Sum / K).

```
15 import random_# Using Python's Random Function
16
17 Sum = 0_# Initialize count of number of favorable outcomes. (accumulator variable).
18
19
20 months = [0, 0, 0]_# Identifies which month is chosen.
21
22 K = 1000_# The number of experiments.
23
24 for k in range(K):_# Total number of experiments to be performed.
25
26     # -----
27     # The experiment starts.
28     # -----
29     for i in range(3):
30         month_Number = random.randrange(1,13)
31         months[i] = month_Number
32     # The experiment ends.
33     # -----
34     # Test for favorable outcomes.
35     if (i == 2)&((months[0] != months[1]) & (months[1] != months[2]) & (months[0] != months[2])):
36         Sum = Sum + 1_# Increment the count of favorable outcomes.
37     # -----
38     # The ratio of favorable outcomes to the total number of outcomes.
39     print(Sum / K)
40
```

The Python Console on the right shows the execution of the script. It displays the values of variables at each step: Can = (list: 3) [1, 9, 3], Can_Number = (int) 3, K = (int) 1000, Sum = (int) 754, i = (int) 2, k = (int) 999, month_Number = (int) 6, months = (list: 3) [10, 1, 6], and Special Variables.

```
> Can = (list: 3) [1, 9, 3]
> Can_Number = (int) 3
> K = (int) 1000
> Sum = (int) 754
> i = (int) 2
> k = (int) 999
> month_Number = (int) 6
> months = (list: 3) [10, 1, 6]
> Special Variables
```