DS1EDP: Homework 05 – Solutions

1. Probability

Question 1:

```
ten_sixes = 2
```

Question 2:

```
five_or_less = 3
```

Question 3:

```
lottery = 3
```

Question :

```
list_chances = 2
```

2. Monkeys Typing Shakespeare

Question 1:

```
datascience_chance = (1/26)**11
```

Question 2:

```
def simulate_key_strike():
    """Simulates one random key strike."""
    return np.random.choice(letters)
```

Question 3:

```
def simulate_several_key_strikes(num_strikes):
    a = make_array()
    for i in range(num_strikes):
        a = np.append(a, simulate_key_strike())
    return "".join(a)
```

Question 4:

```
a = make_array()
for i in range(1000):
    a = np.append(a, simulate_several_key_strikes(11))
datascience_proportion = np.count_nonzero(a == "datascience") / 1000
```

Question 5:

```
good_approach = False
```

Question 6:

```
e_chance = 1 - ((25/26)**11)
```

Question 7:

```
more_effective = True
```

3. Sampling Basketball Players

Question 1:

```
full_data = player_data.join("Name", salary_data, "PlayerName")
```

Question 2:

```
full_data_with_value = full_data.with_column("Value",
full_data.column("Points") / (full_data.column("Salary") / 1000))
full_data_with_value.hist("Value", bins=np.arange(0.0, 1.0, 0.1))
```

4. Earthquakes

Question 1:

```
better_sample = 2
```

Question 2:

```
representative_sample = earthquakes.sample(500)
representative_mean = np.mean(representative_sample.column('mag'))
```

Question 3:

```
maximums = make_array()
for i in np.arange(5000):
    maximums = np.append(maximums,
max(earthquakes.sample(500).column("mag")))
```

Question 4:

```
strongest_earthquake_magnitude = max(earthquakes.column("mag"))
```

Question 5:

```
determining_max_by_sampling = "No"
```

5. Assessing Gary's Models

Question 1:

```
observated_head_probability = 0.1
expected_head_probability = 0.5
coin_model_probabilities = [0.5, 0.5]
```

Question 2:

```
def test_statistic(expected_p, observed_p):
    return abs(expected_p - observed_p)

observed_test_statistic = test_statistic(expected_head_probability,
observated_head_probability)
```

Question 3:

```
def coin_simulation_and_statistic(expected_probability,
model_probabilities):
    x = sample_proportions(10, model_probabilities).item(0)
    return test_statistic(expected_probability, x)

coin_simulation_and_statistic(expected_head_probability,
coin_model_probabilities)
```

Question 4:

```
coin_statistics = make_array()
repetitions = 5000

for i in range(5000):
    coin_statistics = np.append(coin_statistics,
coin_simulation_and_statistic(expected_head_probability,
coin_model_probabilities))
```

Question 5:

```
p_value = np.count_nonzero(coin_statistics >= 0.4) / 5000
print(p_value)
null_hypothesis_rejected = True
```