COSC2429_Assignment1_2021C

1. Find the 80% Split Number

Objective: To find a number in a list of 20 integers where at least 80% of the numbers are equal to or smaller than it.

- **Input**
  - List of 20 integers.
- **Output**
  - A single integer from the list.

Steps:

1. **Understand the List**: A list in programming is a collection of items (in this case, integers) that are ordered and changeable.
2. **Sort the List**: To find the 80% split number, you first need to sort the list in ascending order.
3. **Find the Index**: Calculate the index corresponding to 80% of the length of the list. In a list of 20 items, this would be the 16th item (since 80% of 20 is 16).
4. **Retrieve the Number**: Fetch the number at this index from the sorted list.
5. **Return the Number**: This number is the one where 80% of the numbers in the list are equal to or less than it.

```
def find_split_80(integer_list):
    if not integer_list or len(integer_list) != 20:
        raise ValueError("The list must contain exactly 20 integers.")

    sorted_list = sorted(integer_list)
    # Finding the number where 80% of numbers are equal to or smaller than it
    index = int(len(sorted_list) * 0.8) - 1
    return sorted_list[index]
```

2. Estimate π with Random Points

Objective: To estimate the value of π by generating random points and checking how many fall inside a unit circle.the value of π.

- **Input**
  - Number of random points to generate.
- **Output**
  - Estimated value of π.

Steps:

1. **Understand the Concept**: Imagine a square with a circle inside it. The ratio of the area of the circle to the square can be used to estimate π.
2. **Generate Points**: Use a random number generator to create points within the square.
3. **Check Points Inside Circle**: For each point, check if it lies inside the circle (distance from center ≤ radius, which is 1).
4. **Calculate π**: The ratio of points inside the circle to the total points, multiplied by 4, gives an estimate of π.
5. **Return the Estimate**: The calculated value is your estimated π.

```python
import random
import math

def estimate_pi(num_points):
    """
    Estimates the value of pi using the Monte Carlo method.

    Parameters:
    num_points (int): The number of random points to generate.

    Returns:
    float: The estimated value of pi.
    """

    # Initialize the count of points inside the circle
    points_inside_circle = 0

    # Generate points and count how many fall inside the unit circle
    for _ in range(num_points):
        x, y = random.uniform(-1, 1), random.uniform(-1, 1)  # Generate random x, y coordinates
        distance = math.sqrt(x**2 + y**2)  # Calculate the distance from the origin
        if distance <= 1:
            points_inside_circle += 1  # Point is inside the circle

    # Calculate the estimated pi
    estimated_pi = 4 * points_inside_circle / num_points

    return estimated_pi

# Example usage:
# Estimate pi using 1,000,000 random points
estimated_pi = estimate_pi(1000000)
estimated_pi
```

```
3.142856
```

3. Calculate Flour Order for Pizzas

[ ] ↳ *3 cells hidden*

4. Draw a Stacked Bar Chart with Turtle

↳ *2 cells hidden*