18/9/25-Functions in `random` Module

Example 1: random()

📌 Returns a **floating-point number** between `0.0` and `1.0`.

```
import random

num = random.random()
print("Random number between 0 and 1:", num)
```

🔹 **Explanation**:

- Generates a pseudo-random decimal (float).
- Always in range `[0.0, 1.0)`.

🔹 **Output (example):**

```
Random number between 0 and 1: 0.7365891444
```

Example 2: uniform(a, b)

📌 Returns a random **float between `a` and `b`**.

```
num = random.uniform(5, 10)
print("Random float between 5 and 10:", num)
```

🔹 **Explanation**:

- Useful when you want random decimal values in a range.
- Unlike `randint`, it can give decimals.

🔹 **Output (example):**

```
Random float between 5 and 10: 7.824329
```

Example 3: randint(a, b)

📌 Returns a **random integer between a and b** (inclusive).

```
num = random.randint(1, 6)
print("Random integer between 1 and 6:", num)
```

🔹 **Explanation**:

- Equivalent to rolling a dice 🎲.
- Both end values are included.

🔹 **Output (example):**

```
Random integer between 1 and 6: 4
```

Example 4: randrange(start, stop, step)

📌 Returns a random number from a range.

```
num = random.randrange(1, 10, 2)
print("Random odd number between 1 and 10:", num)
```

🔹 **Explanation**:

- Works like `range(start, stop, step)`.
- Picks a random element from that sequence.

🔹 **Output (example):**

```
Random odd number between 1 and 10: 7
```

Example 5: choice(seq)

📌 Returns a random element from a list, tuple, or string.

```
fruits = ["apple", "banana", "cherry", "mango"]
```

```
print("Random fruit:", random.choice(fruits))
```

🔹 **Explanation**:

- Selects one random element from a sequence.
- Commonly used in games or quizzes.

🔹 **Output (example):**

```
Random fruit: banana
```

Example 6: choices(seq, k=n)

📌 Returns a list of `k` random elements (with replacement).

```
colors = ["red", "blue", "green", "yellow"]
print("Random 3 colors:", random.choices(colors, k=3))
```

🔹 **Explanation**:

- Unlike `choice()`, it can pick multiple items.
- Items can repeat (sampling **with replacement**).

🔹 **Output (example):**

```
Random 3 colors: ['blue', 'green', 'blue']
```

Example 7: sample(seq, k=n)

📌 Returns a list of `k` unique random elements (without replacement).

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]
print("Random 4 numbers:", random.sample(numbers, 4))
```

🔹 **Explanation**:

- Picks `k` unique values.
- No duplicates.

🔹 **Output (example):**

```
Random 4 numbers: [3, 8, 1, 6]
```