■ Python Random Module – Developer Reference Guide

random.randint(a, b)

Returns a random integer N such that a <= N <= b.

■ Use Case: Dice games, token generation, CAPTCHA

random.random()

Returns a float number in the range [0.0, 1.0).

■ Use Case: Probabilistic simulations, random scaling

random.uniform(a, b)

Returns a random floating-point number N such that a <= N <= b.

■ Use Case: Temperature simulator, range-based predictions

random.choice(seq)

Returns a single random element from a non-empty sequence.

■ Use Case: Quiz questions, random prompts

random.choices(seq, k=n)

Returns a list of n random elements from a sequence, with replacement.

■ Use Case: Lottery, random team assignments

random.sample(seq, k)

Returns k unique elements from a sequence (no replacement).

■ Use Case: Lucky draw, sample testing

random.shuffle(lst)

Shuffles a list in place (no return).

■ Use Case: Card games, random ordering

random.seed(n)

Sets the seed for reproducibility of random operations.

■ Use Case: Debugging simulations, experiments

random.getrandbits(k)

Returns a Python integer with k random bits.

■ Use Case: Cryptographic tokens, binary randomness

random.betavariate(alpha, beta)

Beta distribution for modeling probabilities between 0 and 1.

■ Use Case: Bayesian modeling, A/B testing

random.gauss(mu, sigma)

Returns a float based on a Gaussian distribution.

■ Use Case: Height distribution, sensor noise simulation

random.expovariate(lambd)

Exponential distribution useful for time-based events.

■■ Use Case: Queue systems, service times

random.triangular(low, high, mode)

Triangular distribution defined by low, high, and peak mode.

■ Use Case: Risk modeling, delivery time predictions