**📘 TỔNG HỢP LÝ THUYẾT & HƯỚNG DẪN TỪ BÀI 1 → BÀI 7**

**Chủ đề:** Ước lượng tham số thống kê bằng mô phỏng Python  
**Công cụ:** numpy, matplotlib, scipy.stats

**🧩 BÀI 1 – TẠO QUẦN THỂ MÔ PHỎNG**

**🎯 Mục tiêu**

Tạo 10.000 dữ liệu mô phỏng chiều cao nam thanh niên (120–200 cm), tính:

* Trung bình (kỳ vọng)
* Độ lệch chuẩn
* Tỷ lệ người cao (≥180 cm)
* Vẽ histogram và so sánh với phân phối chuẩn

**🧮 Công thức**

**💻 Code chính**

import numpy as np

import matplotlib.pyplot as plt

from scipy.stats import norm

N = 10000

POP = np.random.randint(120, 200, N) # sinh dữ liệu ngẫu nhiên từ 120–199

mean = POP.mean()

std = POP.std()

tyle\_cao = (POP >= 180).sum() / N

plt.hist(POP, bins=30, edgecolor='black', density=True)

x = np.linspace(mean - 4\*std, mean + 4\*std, 100)

y = norm.pdf(x, loc=mean, scale=std)

plt.plot(x, y, color='red')

plt.show()

**🔍 Giải thích hàm**

| **Lệnh** | **Chức năng** | **Tham số chính** |
| --- | --- | --- |
| np.random.randint(a,b,N) | Sinh N giá trị ngẫu nhiên trong [a,b) | a=120, b=200 |
| .mean(), .std() | Tính trung bình và độ lệch chuẩn | — |
| plt.hist(data, bins, density) | Vẽ histogram | bins=30 số cột, density=True chuẩn hóa |
| norm.pdf(x, loc, scale) | Hàm mật độ chuẩn (Normal PDF) | loc=μ, scale=σ |

**🧩 BÀI 2 – LẤY MẪU CỠ 20**

**🎯 Mục tiêu**

Lấy ngẫu nhiên 20 người từ quần thể, tính:

* Trung bình mẫu
* Độ lệch chuẩn mẫu
* Tỷ lệ người cao
* So sánh histogram mẫu với phân phối chuẩn

**💻 Code chính**

sam = np.random.choice(POP, size=20, replace=False)

tb\_mau = sam.mean()

dlc\_mau = sam.std()

tyle\_cao = (sam >= 180).sum() / 20

**🔍 Giải thích**

| **Hàm** | **Ý nghĩa** | **Tham số** |
| --- | --- | --- |
| np.random.choice(a, size, replace) | Lấy ngẫu nhiên size phần tử từ mảng a | replace=False → không lặp |
| .std() | Tính độ lệch chuẩn mẫu | — |

**🧠 Nhận xét**

* Mẫu nhỏ → histogram gồ ghề, dao động lớn.
* Không mang dạng chuẩn nếu quần thể không chuẩn.

**🧩 BÀI 3 – LẤY MẪU CỠ 100**

**🎯 Mục tiêu**

Lặp lại bài 2 với kích thước mẫu lớn hơn (100 người).

**📈 Kết quả**

* Histogram mượt hơn, trung bình mẫu gần trung bình quần thể hơn.
* Minh họa **luật số lớn (Law of Large Numbers)**.

**🧩 BÀI 4 – KHẢO SÁT PHÂN PHỐI CỦA TRUNG BÌNH MẪU**

**🎯 Mục tiêu**

Thực hiện 100 lần lấy mẫu và quan sát **phân phối trung bình mẫu**:

* Với n = 20
* Với n = 100

**💻 Code chính**

def lay\_trung\_binh\_mau(POP, sample\_size, so\_lan=100):

ds\_trung\_binh = []

for i in range(so\_lan):

sam = np.random.choice(POP, sample\_size)

ds\_trung\_binh.append(sam.mean())

return ds\_trung\_binh

means\_20 = lay\_trung\_binh\_mau(POP, 20)

means\_100 = lay\_trung\_binh\_mau(POP, 100)

**🔍 Ý nghĩa**

* Khi n tăng, phân phối trung bình mẫu **gần chuẩn hơn** (theo CLT).
* Trung bình mẫu dao động quanh trung bình quần thể.

**🧩 BÀI 5 – KHẢO SÁT TỶ LỆ MẪU**

**🎯 Mục tiêu**

Mô phỏng **1000 lần lấy mẫu**, mỗi lần tính **tỷ lệ người cao (≥180 cm)**.

**💻 Code chính**

def lay\_ty\_le\_cao(POP, sample\_size, so\_lan=1000):

ds\_tyle = []

for \_ in range(so\_lan):

sam = np.random.choice(POP, size=sample\_size, replace=False)

tyle = (sam >= 180).sum() / sample\_size

ds\_tyle.append(tyle)

return ds\_tyle

**🧠 Nhận xét**

* Khi n tăng → histogram tỷ lệ hẹp lại.
* Độ biến thiên giảm → tỷ lệ mẫu **ổn định hơn**, gần tỷ lệ thật của quần thể.

**🧩 BÀI 6 – PHƯƠNG SAI MẪU**

**🎯 Mục tiêu**

Mô phỏng 1000 lần, mỗi lần lấy mẫu 100 và tính **phương sai**:

**💻 Code chính**

def lay\_phuong\_sai\_mau(POP, sample\_size=100, so\_lan=1000):

ds\_var = []

for \_ in range(so\_lan):

sam = np.random.choice(POP, sample\_size)

ds\_var.append(sam.std()\*\*2)

return ds\_var

**🧠 Kết luận**

* Phân phối phương sai mẫu **lệch phải**, không đối xứng.
* Dạng tương tự **phân phối Chi-bình phương (χ²)**.
* Khi n tăng → phương sai mẫu ổn định hơn.

**🧩 BÀI 7 – ƯỚC LƯỢNG TRUNG BÌNH**

**🎯 Mục tiêu**

Ước lượng **trung bình cân nặng gà** (kg) với mức tin cậy 95%.

**🧮 Công thức**

| **Trường hợp** | **Công thức** | **Phân phối** |
| --- | --- | --- |
| Biết σ |  | Z (chuẩn) |
| Không biết σ |  | t Student |

**💻 Code chính**

from scipy.stats import norm, t

x = np.array([2.1, 1.8, 2.0, 2.3, 1.7, 1.5, 2.0, 2.2, 1.8])

n = len(x)

xbar = x.mean()

s = x.std(ddof=1)

# a) Biết σ

sigma = 0.3

z\_crit = norm.ppf(0.975)

se\_z = sigma / np.sqrt(n)

ci\_z = (xbar - z\_crit\*se\_z, xbar + z\_crit\*se\_z)

# b) Không biết σ

t\_crit = t.ppf(0.975, df=n-1)

se\_t = s / np.sqrt(n)

ci\_t = (xbar - t\_crit\*se\_t, xbar + t\_crit\*se\_t)

**🔍 Giải thích hàm**

| **Hàm** | **Chức năng** | **Tham số** |
| --- | --- | --- |
| norm.ppf(0.975) | Trả về giá trị z sao cho P(Z<z)=0.975 | 95% độ tin cậy |
| t.ppf(0.975, df) | Giá trị t tới hạn | df = n-1 |
| .std(ddof=1) | Độ lệch chuẩn mẫu | ddof=1 để dùng (n−1) |
| np.sqrt(n) | Căn bậc hai của kích thước mẫu | — |

**📊 Kết quả ví dụ**

x̄ = 1.933, s = 0.262

Biết σ=0.3: CI = (1.737, 2.129)

Không biết σ: CI = (1.732, 2.134)

**🧠 Nhận xét**

* Khoảng tin cậy t rộng hơn một chút → phản ánh **bất định cao hơn** khi không biết σ.
* Khi n tăng, phân phối t tiến gần phân phối Z.

**📘 TỔNG HỢP KIẾN THỨC CỐT LÕI**

| **Chủ đề** | **Ký hiệu** | **Công thức** | **Ý nghĩa** |
| --- | --- | --- | --- |
| Trung bình |  |  | Đại diện xu hướng trung tâm |
| Độ lệch chuẩn |  |  | Độ phân tán dữ liệu |
| Phương sai |  |  | Mức độ biến thiên bình phương |
| Sai số chuẩn |  | hoặc | Độ dao động của trung bình |
| Khoảng tin cậy | CI | hoặc | Khoảng ước lượng trung bình |
| Định lý giới hạn trung tâm | — | Trung bình mẫu có phân phối gần chuẩn khi n lớn | Cơ sở cho ước lượng và kiểm định |