I. Cách chạy mô hình

1. Thay đổi giá trị của biến "`TRAIN_DATA_DIR_PATH`" ở dòng 24 của file *main.py* thành địa chỉ chứa thư mục train của hệ thống
2. Thay đổi giá trị của biến "`TEST_DATA_DIR_PATH`" ở dòng 25 của file *main.py* thành địa chỉ chứa thư mục test của hệ thống
3. Tải các thư mục cần thiết (CLI phải ở trong thư mục chứa file *requirements.txt*):
```
pip install -r requirements.txt
```
4. Chạy chương trình với lệnh:
```
python main.py
```

II. Ý tưởng mô hình

1. Mô hình sử dụng: ConvNeXt-Tiny

- ConvNeXt-Tiny là một kiến trúc CNN hiện đại, lấy cảm hứng từ cả CNN truyền thống và các ý tưởng từ Transformer, tối ưu hóa cho bài toán phân loại ảnh.
- Trong code, mô hình này được khởi tạo từ torchvision.models.convnext_tiny với trọng số pretrained (pretrained=True), giúp tận dụng kiến thức từ tập dữ liệu lớn (ImageNet).
- Lớp phân loại cuối cùng (classifier[2]) được thay thế bằng một lớp Linear với số lượng đầu ra đúng bằng số lớp của bài toán (ở đây là 4 lớp nấm).

2. Quy trình (pipeline) giải quyết bài toán

a. Cấu hình augmentation

```
resize: 224
horizontal_flip: 0.5
vertical_flip: 0.5
random_rotation: 90
random_affine: 0.2
random_perspective: 0.2
edge_enhance: true
```

```
auto_aug: 1
```

b. Cấu hình mô hình và hàm tối ưu

```
scheduler: "cosine"

#*-- hyperparameters
num_classes: 4
lr: 1e-5
num_epochs: 200
batch_size: 16
loss: "cross_entropy"
is_weight: false
optimizer: "adamw"
weight_decay: 0.00025
momentum: 0.9
```

c. Chia tập dữ liệu

Tập train được chia thành train/val theo tỷ lệ 90/10 bằng random_split.

d. Huấn luyện

Lặp qua các epoch:
Train: Forward, tính loss, backward, update weight.
Validation: Đánh giá loss và accuracy trên tập val.
Checkpoint: Lưu lại mô hình tốt nhất (theo val loss).

e. Inference & Xuất kết quả

Load lại mô hình tốt nhất.
Duyệt qua từng ảnh test, dự đoán nhãn, lưu kết quả ra file CSV.
Tóm tắt:

- **Mô hình**: ConvNeXt-Tiny (CNN hiện đại, pretrained, fine-tune cho bài toán nấm).
- **Pipeline**: Đọc cấu hình → Augmentation → Chia tập → Xây dựng mô hình → Chọn optimizer/loss → Huấn luyện với Checkpoint → Inference → Xuất kết quả.
- **Mục tiêu**: Phân loại ảnh nấm thành 4 lớp, tối ưu hóa độ chính xác trên tập kiểm thử.