```python
import requests  # grabbing the requests module so we can make HTTP calls

"""
Ticket Handling Script
This script fakes a support ticket system using a mock API.
It can create, show, and "update" tickets (not real ones though).
"""

# This is the fake API we're using (JSONPlaceholder)
TICKET_ENDPOINT = "https://jsonplaceholder.typicode.com/posts"

def create_ticket(issue):
    """Pretend to make a new support ticket."""
    data = {
        "title": "Network Outage",  # Just a default title we're using
        "body": issue,              # The actual issue we're passing in
        "userId": 1                 # Random user ID (doesn't really matter here)
    }
    response = requests.post(TICKET_ENDPOINT, json=data)  # sending a POST request with our ticket data
    return response.json()  # get the response back in JSON so we can use it

def list_tickets(ticket):
    """Show the ticket we just 'created'."""
    return [ticket]  # wrapping the ticket in a list to make it look like a list of tickets

def update_ticket(ticket_id, new_status):
    """Fake updating a ticket with a new status."""
    print(f"✅ Ticket #{ticket_id} updated: marked as {new_status}")  # just printing the update
    return {"ticket_id": ticket_id, "status": new_status}  # returning a fake update response

# This is where the script actually runs
if __name__ == "__main__":  # make sure we're running this directly
    print("Creating ticket...")  # tell the user what’s going on
    ticket = create_ticket("Switch down in Lab A")  # make a new ticket with a specific issue
    print(ticket)  # print the ticket info

    print("\nListing ticket...")  # now we list the ticket we just made
    for t in list_tickets(ticket):  # go through the (single) ticket
        print(f"Ticket {t['id']}: {t['title']}")  # print the ticket ID and title

    print("\nUpdating ticket...")  # time to "update" the ticket
    updated = update_ticket(ticket['id'], "closed")  # close the ticket
    print(updated)  # print the fake update result
```