Question 3

Application Protocol

1. **Client to Server Messages:**
    - **Request to Display Catalog:**
        - Message Type: 1
        - Client sends: 1
        - Server responds:
            - Number of books (count)
            - Client sends: Number of books client wants to receive (book_count)
            - Server sends: Details of each book (up to book_count times)
    - **Request to Search Book:**
        - Message Type: 2
        - Client sends: 2, Search Type (1 for title, 2 for ISBN), Search Parameter (title or ISBN)
        - Server responds:
            - Status ("found" or "not found")
            - If found, details of the book
    - **Request to Order Book:**
        - Message Type: 3
        - Client sends: 3, ISBN of the book to order
        - Server responds:
            - Status ("found" or "not found")
            - If found, order number for the book
    - **Request to Pay for Order:**
        - Message Type: 4
        - Client sends: 4, Order Number to pay for
        - Server responds:
            - Status ("found" or "not found")
            - If found, confirmation of payment
    - **Exit Request:**
        - Message Type: 5
        - Client sends: 5
        - Server terminates the connection and exits
2. **Server to Client Messages:**
    - **Sending Data:**
        - Server sends data based on the client's request:
            - Number of books (count)
            - Book details (book structure)
            - Search status ("found" or "not found")
            - Order status ("unpaid" or "paid")
            - Order number

Question 3

Conceptual Server Algorithm

The server operates in a loop, continuously listening for client requests over UDP. Below is a conceptual algorithm outlining how the server handles incoming requests and manages interactions with the client:

1. **Initialize Server:**
    - Create a UDP socket (socket()).
    - Bind socket to a specific port (bind()).
2. **Server Main Loop:**
    - Continuously listen for incoming client requests (recvfrom()).
3. **Handle Client Request (handle_client()):**
    - Receive the client's choice of action (e.g., display catalog, search book, order book, pay for order, exit).
    - Based on the choice, call corresponding functions (display_catalog(), search_book(), order_book(), pay_for_book()).
    - Respond to the client with appropriate data or status.
4. **Display Catalog Function (display_catalog()):**
    - Read book details from bookfile.txt.
    - Send the total number of books (count) to the client.
    - Receive the number of books client wants to receive (book_count).
    - Send details of each book requested by the client.
5. **Search Book Function (search_book()):**
    - Read book details from bookfile.txt.
    - Receive search type (title or ISBN) and search parameter from the client.
    - Search for the book based on the provided criteria.
    - Send search status ("found" or "not found") and book details if found.
6. **Order Book Function (order_book()):**
    - Read book details from bookfile.txt.
    - Receive ISBN of the book to order from the client.
    - Search for the book by ISBN.
    - If found, generate a unique order number and append order details to orderfile.txt.
    - Send order status ("found" or "not found") and order number if successful.
7. **Pay for Book Function (pay_for_book()):**
    - Read order details from orderfile.txt.
    - Receive order number from the client.
    - Search for the order by order number.
    - If found, mark the order status as "paid" in orderfile.txt.
    - Send payment status ("found" or "not found") and confirmation of payment if successful.
8. **Exit Functionality:**
    - If the client requests to exit (5), gracefully terminate the server.
9. **Error Handling:**
    - Handle errors such as file not found, socket errors, and unexpected client requests.
    - Provide appropriate error messages or terminate the server if critical errors occur.