Objective: This assignment will deepen your understanding of the fork() and pipe() system calls by requiring you to orchestrate communication between three related processes.

Problem Statement: Develop a C program that performs the following steps:

1. The main (parent) process creates a pipe.
2. The parent then forks two children. The first child created will be Child A, and the second will be Child B.
3. Child A will fork its own child, Child C.
4. The three children will use the parent's pipe for communication as follows:
    - **Child B:** This process will read a predefined string from the user via standard input (stdin).
    - **Child A:** This process will read the string that Child B received from standard input and then send it to Child C through the pipe.
    - **Child C:** This process will read the message from the pipe and print it to standard output (stdout).
5. All parent processes (main and Child A) must wait for their respective children to terminate.

Requirements:

- **Process Creation:** The main parent process must create Child A and Child B. Child A must then create Child C.
- **Inter-Process Communication:**
    - Child B must read a string from the user.
    - The main process must be a parent to both Child A and Child B. The message should be sent from Child B to Child A through standard input (stdin).
    - The message should be then sent from Child A to Child C through the pipe.
- **Pipe and File Descriptor Management:**
    - The main parent process should close both the read and write ends of the pipe after forking its children.
    - Child A should close the write end of the pipe after it sends the message to Child C.
    - Child B should not interact with the pipe. It will only read from standard input (stdin).
    - Child C should close the write end of the pipe after it receives the message from Child A.
- **Synchronization:** The main parent process must wait for both Child A and Child B to terminate. Child A must wait for Child C to terminate.

- **Error Handling:** All system calls must be checked for errors. Use perror() for informative error messages.

Submission Deliverables:

1. A single C file named three_pipe_comm.c containing your complete, well-commented code.
2. A screenshot of your terminal showing the compilation and a successful run of the program. The screenshot must clearly show the user's input and the program's output.
3. A short text document or a comment block in your code that answers the following question:
    - What type of pipe is created by the pipe() system call? Explain your reasoning and how it facilitates the one-way communication in this assignment.
    - What type of exec() did you use for the three child processes? Why?
    - What type of wait() did main and Child A used? Why?