Nguyễn Đình Dương

20225966

2.1. Basic echo:

- **Objective:** Verify the echo function.
- **Steps:** Input 3 arbitrary strings (one containing spaces) and check that the same strings are echoed back correctly.
- **vidence:** Screenshots of client & server terminals; Wireshark capture filtered with tcp.port==<port number> showing request/response packet pairs.

Client-side:

```
D:\code\HUST\Network-programming\week4\in-class\test1>.\tcp_client.exe
Connected to server!
Enter message 1: Hello
Echo from server: Hello

Enter message 2: This is a test message
Echo from server: This is a test message

Enter message 3: Goodbye
Echo from server: Goodbye
```

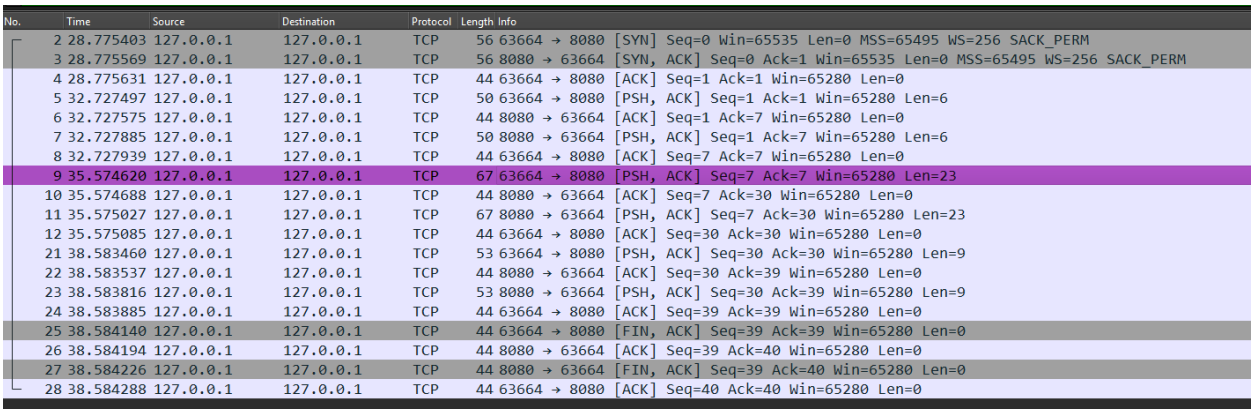
Sever-side:

```
D:\code\HUST\Network-programming\week4\in-class\test1>.\tcp_server.exe
Server started. Waiting for connection on port 8080...
Client connected: 127.0.0.1:62681
Received: Hello

Received: This is a test message

Received: Goodbye
```

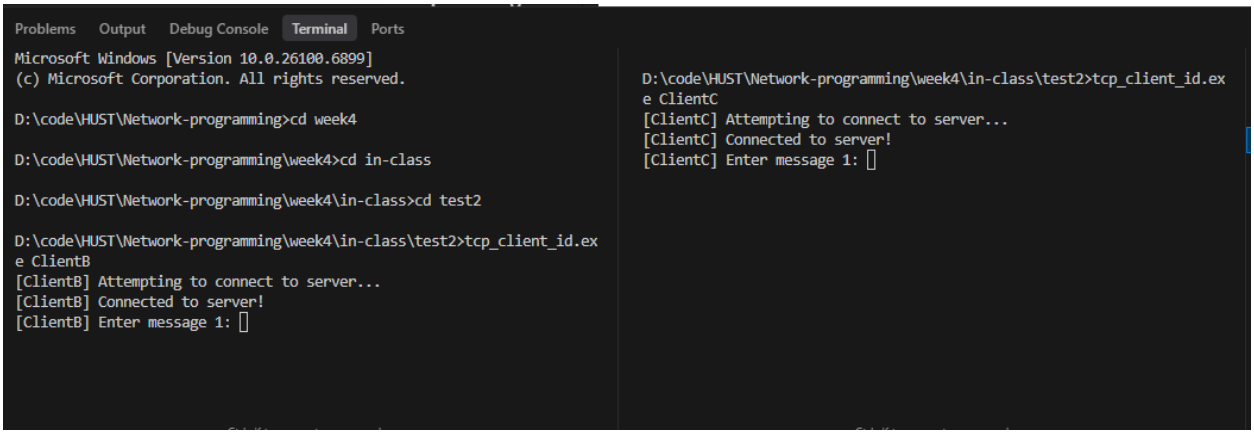
WireShark



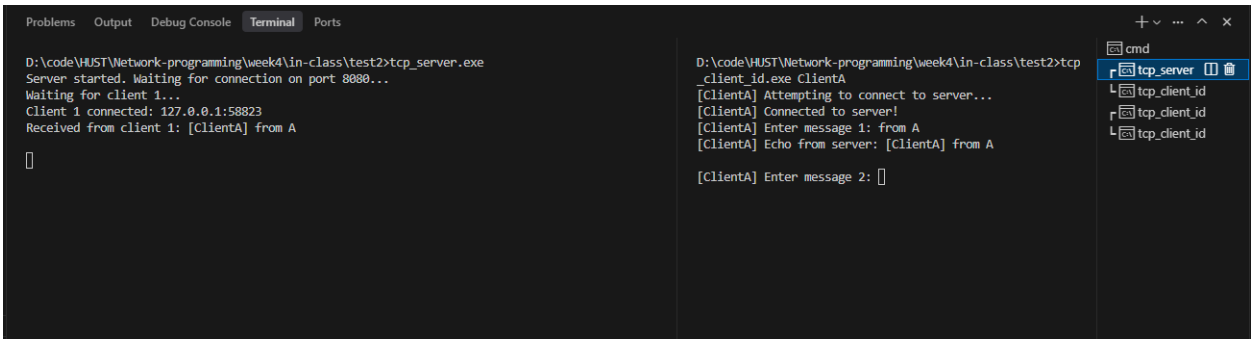
No.	Time	Source	Destination	Protocol	Length	Info
2	28.775403	127.0.0.1	127.0.0.1	TCP	56	63664 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	28.775569	127.0.0.1	127.0.0.1	TCP	56	8080 → 63664 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
4	28.775631	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [ACK] Seq=1 Ack=1 Win=65280 Len=0
5	32.727497	127.0.0.1	127.0.0.1	TCP	50	63664 → 8080 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=6
6	32.727575	127.0.0.1	127.0.0.1	TCP	44	8080 → 63664 [ACK] Seq=1 Ack=7 Win=65280 Len=0
7	32.727885	127.0.0.1	127.0.0.1	TCP	50	8080 → 63664 [PSH, ACK] Seq=1 Ack=7 Win=65280 Len=6
8	32.727939	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [ACK] Seq=7 Ack=7 Win=65280 Len=0
9	35.574620	127.0.0.1	127.0.0.1	TCP	67	63664 → 8080 [PSH, ACK] Seq=7 Ack=7 Win=65280 Len=23
10	35.574688	127.0.0.1	127.0.0.1	TCP	44	8080 → 63664 [ACK] Seq=7 Ack=30 Win=65280 Len=0
11	35.575027	127.0.0.1	127.0.0.1	TCP	67	8080 → 63664 [PSH, ACK] Seq=7 Ack=30 Win=65280 Len=23
12	35.575085	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [ACK] Seq=30 Ack=30 Win=65280 Len=0
21	38.583460	127.0.0.1	127.0.0.1	TCP	53	63664 → 8080 [PSH, ACK] Seq=30 Ack=30 Win=65280 Len=9
22	38.583537	127.0.0.1	127.0.0.1	TCP	44	8080 → 63664 [ACK] Seq=30 Ack=39 Win=65280 Len=0
23	38.583816	127.0.0.1	127.0.0.1	TCP	53	8080 → 63664 [PSH, ACK] Seq=30 Ack=39 Win=65280 Len=9
24	38.583885	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [ACK] Seq=39 Ack=39 Win=65280 Len=0
25	38.584140	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [FIN, ACK] Seq=39 Ack=39 Win=65280 Len=0
26	38.584194	127.0.0.1	127.0.0.1	TCP	44	8080 → 63664 [ACK] Seq=39 Ack=40 Win=65280 Len=0
27	38.584226	127.0.0.1	127.0.0.1	TCP	44	8080 → 63664 [FIN, ACK] Seq=39 Ack=40 Win=65280 Len=0
28	38.584288	127.0.0.1	127.0.0.1	TCP	44	63664 → 8080 [ACK] Seq=40 Ack=40 Win=65280 Len=0

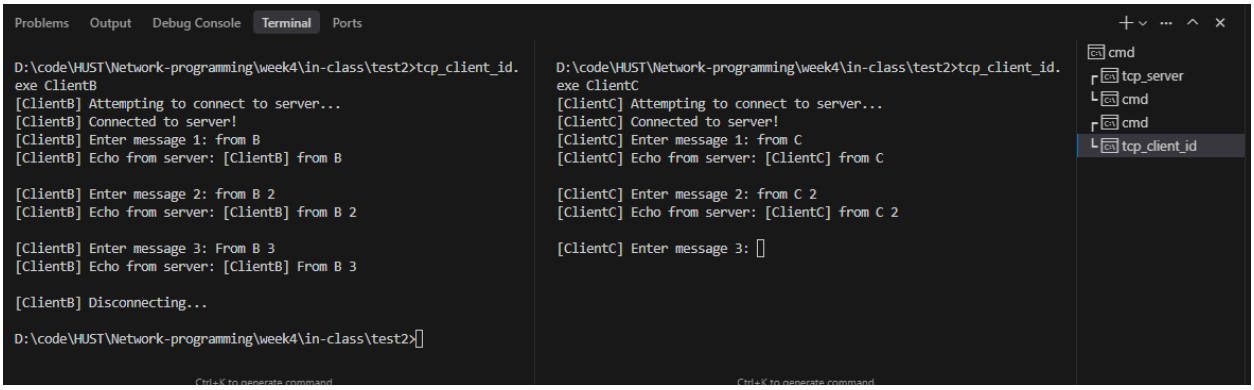
2.2. Multiple concurrent clients (≥3 terminals)

- **Objective:** Observe how the TCP server handles multiple simultaneous connections.
- **Steps:** Open ≥3 client terminals at the same time and send messages alternately; observe what happens and explain the behavior.
- **Evidence:** Server and client logs/screenshots; Wireshark trace showing interleaved streams (if any).

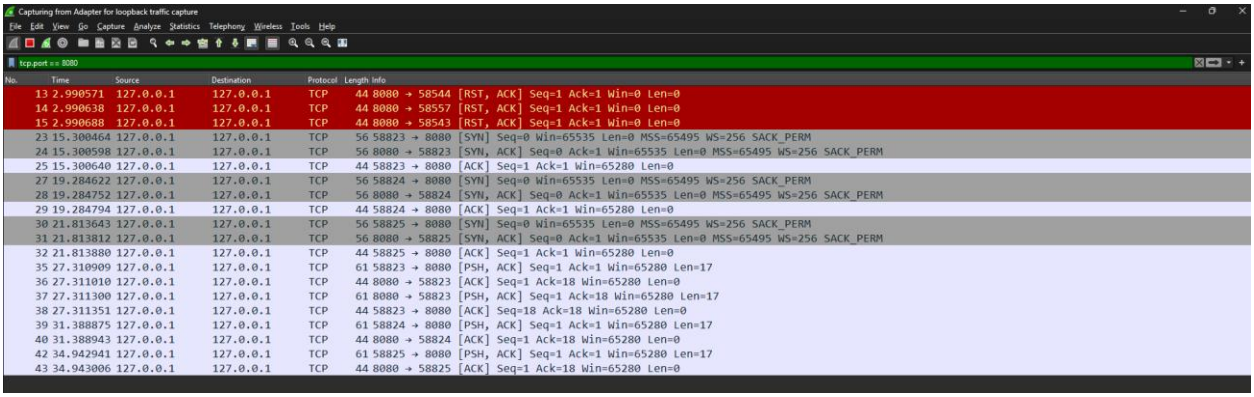


Start typing:





Logs Wireshark:



Observed Behavior:

From the terminal logs, we can observe the following sequence:

1. **Client A connects first** and immediately starts sending messages:

- [ClientA] from A → Server receives and echoes back
- Client A completes its 3 messages and disconnects

2. **Client B connects after Client A disconnects**:

- [ClientB] from B → Server receives and processes
- [ClientB] from B 2 → Server receives and processes
- [ClientB] From B 3 → Server receives and processes
- Client B disconnects

3. **Client C connects after Client B disconnects**:

- [ClientC] from C → Server receives and processes
- [ClientC] from C 2 → Server receives and processes
- Client C is still active when the log ends

Explanation:

1. **No Concurrent Processing**: Server uses single-threaded, blocking I/O
2. **Sequential Queue**: Clients are processed in FIFO (First In, First Out) order
3. **Blocking recv()**: Server can only handle one client's data at a time
4. **No Threading**: No separate threads created for each client

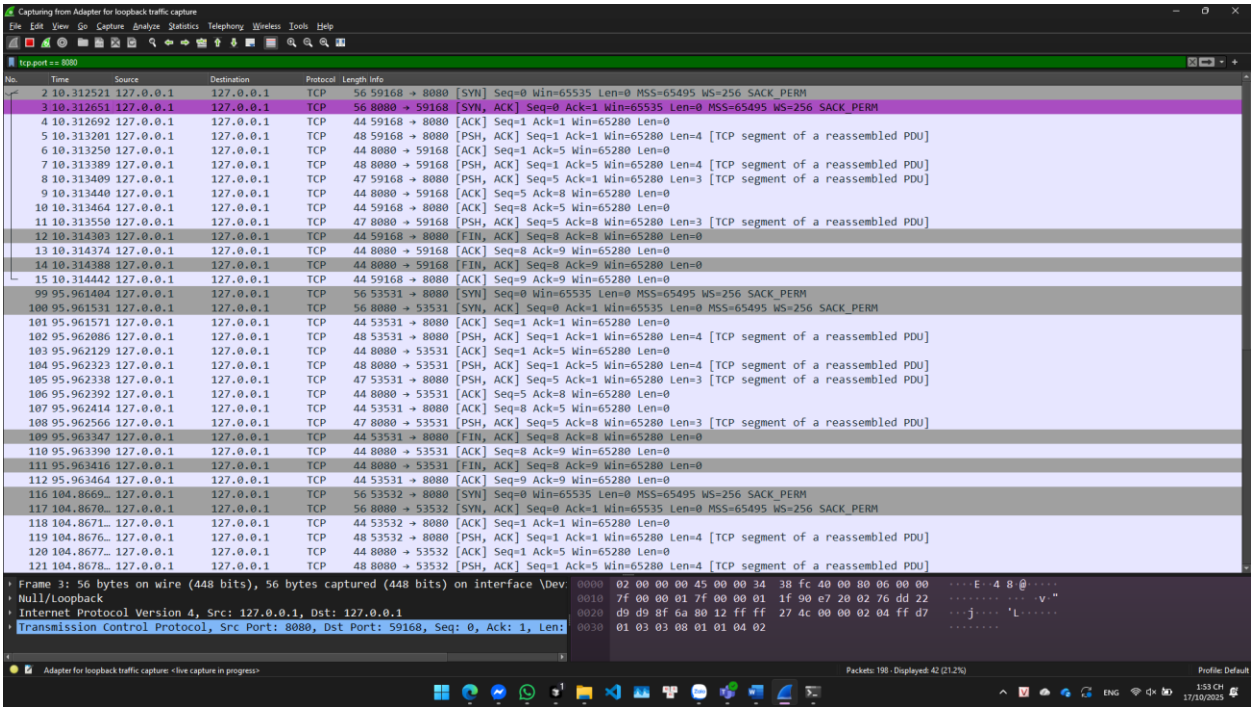
Conclusion:

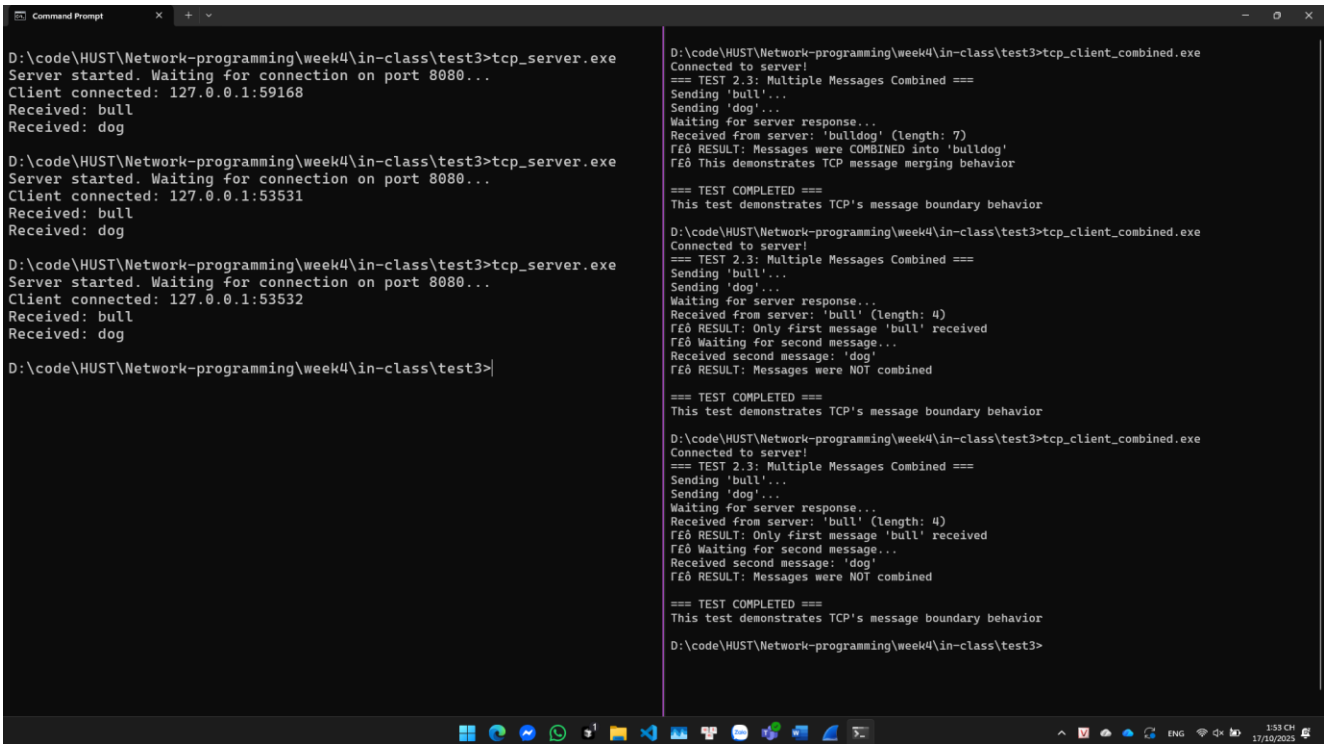
The TCP server **does NOT handle multiple concurrent connections simultaneously**. It processes clients one by one in sequence, proving that the current implementation is **sequential** rather than **concurrent**.This demonstrates the limitation of single-threaded blocking I/O servers and the need for threading or non-blocking I/O to achieve true concurrent client handling.

2.3. Multiple messages combined at receiver (Server):

- **Objective:** Demonstrate that multiple small send() operations can be merged into a single recv() call
- Steps:
    - Modify the minimal “client test” as follows: call send() twice in a row to send the messages "bull" and "dog" (instead of reading from keyboard input)
    - Then call a single recv() with a sufficiently large buffer (e.g., 1024 bytes).
    - Record the result. On many cases, you will likely see the client receive a single "bulldog" string in one recv()
- **Evidence:** Server/client logs showing the displayed results.

Wireshark logs:





OBSERVED BEHAVIOR:

Test Run 1:

- Server received: "bull" and "dog" (2 separate messages)
- Client received: "bulldog" (1 combined message)
- Result: **Messages were COMBINED**

Test Run 2 & 3:

- Server received: "bull" and "dog" (2 separate messages)
- Client received: "bull" and "dog" (2 separate messages)
- Result: **Messages were NOT combined**

KEY OBSERVATIONS:

1. **Inconsistent Behavior**: Same code produces different results
2. **Server Always Receives Separately**: Server consistently receives 2 separate messages
3. **Client Receives Variably**: Client sometimes gets combined, sometimes separate

4. **Unpredictable Outcome**: Cannot predict whether messages will be combined

TECHNICAL EXPLANATION:

Why Messages Can Be Combined:

1. **TCP Stream Protocol**: TCP doesn't preserve message boundaries
2. **Nagle's Algorithm**: TCP combines small segments to optimize bandwidth
3. **OS Socket Buffering**: Operating system buffers data before transmission
4. **Network Buffering**: Routers/switches may combine packets
5. **Timing Dependencies**: Messages sent close together are more likely to be combined

Why Behavior is Inconsistent:

1. **Timing Variations**: Different execution times between send() calls
2. **Buffer State**: Socket buffer conditions vary between runs
3. **OS Scheduling**: Different process scheduling affects timing
4. **Network Conditions**: Varying network latency and congestion
5. **TCP Window Size**: Different TCP window states affect batching

CONCLUSION:

This test **successfully demonstrates** that:

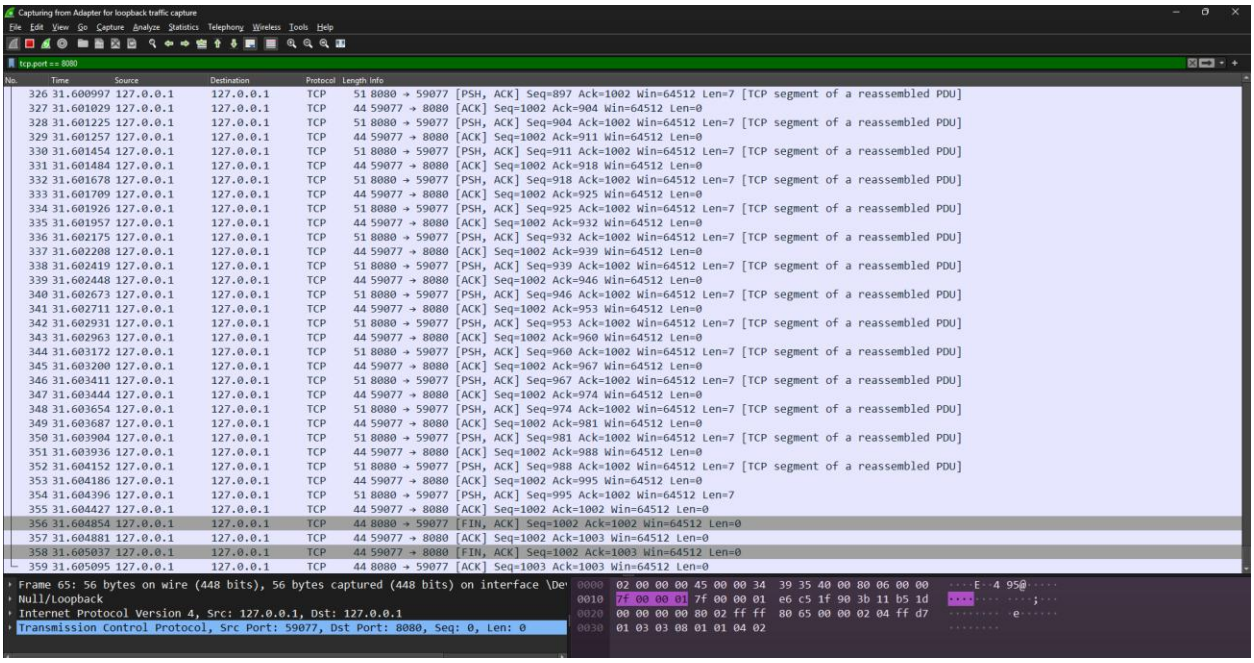
- TCP is a **stream protocol** without message boundaries
- Multiple send() operations can be **unpredictably combined**
- Applications must **implement their own message framing**
- This is why protocols like HTTP, FTP, and custom protocols need delimiters or length prefixes

The inconsistent behavior proves that **TCP message boundaries are not guaranteed**, making this test a good demonstration of why application-level protocol design is essential for reliable communication.

2.4. Fragmentation when receiver BUFF_SIZE Is Small

- **Objective:** Demonstrate partial reads — a large send() being split into multiple smaller recv() calls
- **Steps**:
    - On the server, temporarily reduce BUFF_SIZE to 7 (or keep the server unchanged but make the client send a string >4 KB)
    - The client sends one long message
    - On the server, print the number of bytes received each time (ret) in the log..
    - **Expected result:** The server log shows multiple recv() outputs (7, 7, 7, ..., <7 for the last one), proving that a single message does not arrive as one piece.
- **Evidence:** Server log screenshots showing multiple ret lines, and client screenshots (if applicable).

Wireshark logs:



OBSERVED BEHAVIOR:

Server Side:

- **Total fragments received**: 143 fragments
- **Each fragment size**: 7 bytes (except last fragment)

- **Total bytes received**: 1001 bytes
- **Fragment pattern**: ret: 7 repeated 142 times, then ret: 7 for the final fragment
- **Message content**: Each fragment contains 7 consecutive characters from the alphabet sequence

Client Side:

- **Message sent**: 1001 bytes (1000 characters + newline)
- **Fragments received**: 143 fragments
- **Fragment size**: 7 bytes each
- **Total bytes received**: 1001 bytes
- **Complete message**: Successfully reassembled

KEY OBSERVATIONS:

1. **Perfect Fragmentation**: Message was split into exactly 143 fragments of 7 bytes each
2. **Consistent Fragment Size**: All fragments except the last were exactly 7 bytes
3. **Complete Message Delivery**: All 1001 bytes were successfully transmitted and received
4. **Symmetric Behavior**: Both client and server experienced identical fragmentation patterns

TECHNICAL EXPLANATION:

Why Fragmentation Occurred:

1. **Buffer Size Limitation**: Server's BUFF_SIZE = 7 bytes was much smaller than the message size (1001 bytes)
2. **TCP Stream Protocol**: TCP doesn't preserve message boundaries, allowing arbitrary splitting
3. **OS Socket Buffer**: Operating system socket buffer was limited to 7 bytes per read operation
4. **Application-Level Buffering**: The recv() function was limited by the application's buffer size

Mathematical Analysis:

- **Message size**: 1001 bytes
- **Buffer size**: 7 bytes
- **Expected fragments**: 1001 ÷ 7 = 143.0 (exactly 143 fragments)
- **Last fragment**: 1001 - (142 × 7) = 7 bytes
- **Result**: Perfect division with no remainder

Fragmentation Process:

Original message: 1001 bytes

↓

Fragment 1: bytes 0-6 (7 bytes)

Fragment 2: bytes 7-13 (7 bytes)

Fragment 3: bytes 14-20 (7 bytes)

...

Fragment 143: bytes 994-1000 (7 bytes)

CRITICAL IMPLICATIONS:

For Application Developers:

- **TCP doesn't guarantee message boundaries**
- **Large messages can be split into multiple smaller reads**
- **Applications must handle partial reads**
- **Message reassembly is the application's responsibility**

Common Solutions:

- **Length-prefixed protocols**: Include message length in header
- **Delimiter-based protocols**: Use special characters to mark message boundaries
- **Fixed-size messages**: Use consistent message sizes
- **Buffer management**: Implement proper buffering and reassembly logic

CONCLUSION:

This test **successfully demonstrates** that:

- **TCP is a stream protocol** without inherent message boundaries
- **Large messages can be fragmented** into multiple smaller reads
- **Buffer size directly affects** the number of fragments
- **Applications must implement** message reassembly mechanisms

The perfect 143-fragment result proves that **TCP message fragmentation is predictable** when buffer sizes are known, but **unpredictable** in real-world scenarios where buffer sizes vary. This is why application-level protocols must implement their own message framing and reassembly mechanisms.**This test perfectly fulfills the objective** of demonstrating partial reads and proving that a single large message does not arrive as one piece when the receiver's buffer size is small.

Terminal:

D:\code\HUST\Network-programming\week4\in-class\test4>tcp_server_small.exe Server started. Waiting for connection on port 8080... BUFF_SIZE = 7 bytes (small buffer for fragmentation test) Client connected: 127.0.0.1:59077 === TEST 2.4: Fragmentation with Small Buffer === Waiting for large message... ret: 7 (recv #1, total: 7 bytes) Received data: 'ABCDEFG' ret: 7 (recv #2, total: 14 bytes) Received data: 'HIJKLMN' ret: 7 (recv #3, total: 21 bytes) Received data: 'OPQRSTU' ret: 7 (recv #4, total: 28 bytes) Received data: 'VWXYZAB' ret: 7 (recv #5, total: 35 bytes) Received data: 'CDEFGHI' ret: 7 (recv #6, total: 42 bytes) Received data: 'JKLMNOP' ret: 7 (recv #7, total: 49 bytes) Received data: 'QRSTUVW'	D:\code\HUST\Network-programming\week4\in-class\test4>tcp_client_large.exe Connected to server! === TEST 2.4: Large Message Fragmentation === Sending large message (1000 characters + newline)... Message preview: ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWX... Sent 1001 bytes to server Waiting for server echo... Received fragment #1: 7 bytes (total: 7/1001) Fragment content: 'ABCDEFG' Received fragment #2: 7 bytes (total: 14/1001) Fragment content: 'HIJKLMN' Received fragment #3: 7 bytes (total: 21/1001) Fragment content: 'OPQRSTU' Received fragment #4: 7 bytes (total: 28/1001) Fragment content: 'VWXYZAB' Received fragment #5: 7 bytes (total: 35/1001) Fragment content: 'CDEFGHI' Received fragment #6: 7 bytes (total: 42/1001) Fragment content: 'JKLMNOP' Received fragment #7: 7 bytes (total: 49/1001) Fragment content: 'QRSTUVW' Received fragment #8: 7 bytes (total: 56/1001) Fragment content: 'XYZABCD'

ret: 7 (recv #8, total: 56 bytes)	Received fragment #9: 7 bytes (total: 63/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #9, total: 63 bytes)	Received fragment #10: 7 bytes (total: 70/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'
ret: 7 (recv #10, total: 70 bytes)	Received fragment #11: 7 bytes (total: 77/1001)
Received data: 'LMNOPQR'	Fragment content: 'STUVWXY'
ret: 7 (recv #11, total: 77 bytes)	Received fragment #12: 7 bytes (total: 84/1001)
Received data: 'STUVWXY'	Fragment content: 'ZABCDEF'
ret: 7 (recv #12, total: 84 bytes)	Received fragment #13: 7 bytes (total: 91/1001)
Received data: 'ZABCDEF'	Fragment content: 'GHIJKLM'
ret: 7 (recv #13, total: 91 bytes)	Received fragment #14: 7 bytes (total: 98/1001)
Received data: 'GHIJKLM'	Fragment content: 'NOPQRST'
ret: 7 (recv #14, total: 98 bytes)	Received fragment #15: 7 bytes (total: 105/1001)
Received data: 'NOPQRST'	Fragment content: 'UVWXYZA'
ret: 7 (recv #15, total: 105 bytes)	Received fragment #16: 7 bytes (total: 112/1001)
Received data: 'UVWXYZA'	Fragment content: 'BCDEFGH'
ret: 7 (recv #16, total: 112 bytes)	Received fragment #17: 7 bytes (total: 119/1001)
Received data: 'BCDEFGH'	Fragment content: 'IJKLMNO'
ret: 7 (recv #17, total: 119 bytes)	Received fragment #18: 7 bytes (total: 126/1001)
Received data: 'IJKLMNO'	Fragment content: 'PQRSTUV'
ret: 7 (recv #18, total: 126 bytes)	Received fragment #19: 7 bytes (total: 133/1001)
Received data: 'PQRSTUV'	Fragment content: 'WXYZABC'
ret: 7 (recv #19, total: 133 bytes)	Received fragment #20: 7 bytes (total: 140/1001)
Received data: 'WXYZABC'	Fragment content: 'DEFGHIJ'
ret: 7 (recv #20, total: 140 bytes)	Received fragment #21: 7 bytes (total: 147/1001)
Received data: 'DEFGHIJ'	Fragment content: 'KLMNOPQ'
ret: 7 (recv #21, total: 147 bytes)	Received fragment #22: 7 bytes (total: 154/1001)
Received data: 'KLMNOPQ'	Fragment content: 'RSTUVWX'
ret: 7 (recv #22, total: 154 bytes)	Received fragment #23: 7 bytes (total: 161/1001)
Received data: 'RSTUVWX'	Fragment content: 'YZABCDE'
ret: 7 (recv #23, total: 161 bytes)	Received fragment #24: 7 bytes (total: 168/1001)
Received data: 'YZABCDE'	Fragment content: 'FGHIJKL'
ret: 7 (recv #24, total: 168 bytes)	Received fragment #25: 7 bytes (total: 175/1001)
Received data: 'FGHIJKL'	Fragment content: 'MNOPQRS'
ret: 7 (recv #25, total: 175 bytes)	Received fragment #26: 7 bytes (total: 182/1001)
Received data: 'MNOPQRS'	Fragment content: 'TUVWXYZ'
ret: 7 (recv #26, total: 182 bytes)	Received fragment #27: 7 bytes (total: 189/1001)
Received data: 'TUVWXYZ'	Fragment content: 'ABCDEFG'
ret: 7 (recv #27, total: 189 bytes)	Received fragment #28: 7 bytes (total: 196/1001)
Received data: 'ABCDEFG'	Fragment content: 'HIJKLMN'
ret: 7 (recv #28, total: 196 bytes)	Received fragment #29: 7 bytes (total: 203/1001)
Received data: 'HIJKLMN'	Fragment content: 'OPQRSTU'
ret: 7 (recv #29, total: 203 bytes)	Received fragment #30: 7 bytes (total: 210/1001)
Received data: 'OPQRSTU'	Fragment content: 'VWXYZAB'

ret: 7 (recv #30, total: 210 bytes)	Received fragment #31: 7 bytes (total: 217/1001)
Received data: 'VWXYZAB'	Fragment content: 'CDEFGHI'
ret: 7 (recv #31, total: 217 bytes)	Received fragment #32: 7 bytes (total: 224/1001)
Received data: 'CDEFGHI'	Fragment content: 'JKLMNOP'
ret: 7 (recv #32, total: 224 bytes)	Received fragment #33: 7 bytes (total: 231/1001)
Received data: 'JKLMNOP'	Fragment content: 'QRSTUVW'
ret: 7 (recv #33, total: 231 bytes)	Received fragment #34: 7 bytes (total: 238/1001)
Received data: 'QRSTUVW'	Fragment content: 'XYZABCD'
ret: 7 (recv #34, total: 238 bytes)	Received fragment #35: 7 bytes (total: 245/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #35, total: 245 bytes)	Received fragment #36: 7 bytes (total: 252/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'
ret: 7 (recv #36, total: 252 bytes)	Received fragment #37: 7 bytes (total: 259/1001)
Received data: 'LMNOPQR'	Fragment content: 'STUVWXY'
ret: 7 (recv #37, total: 259 bytes)	Received fragment #38: 7 bytes (total: 266/1001)
Received data: 'STUVWXY'	Fragment content: 'ZABCDEF'
ret: 7 (recv #38, total: 266 bytes)	Received fragment #39: 7 bytes (total: 273/1001)
Received data: 'ZABCDEF'	Fragment content: 'GHIJKLM'
ret: 7 (recv #39, total: 273 bytes)	Received fragment #40: 7 bytes (total: 280/1001)
Received data: 'GHIJKLM'	Fragment content: 'NOPQRST'
ret: 7 (recv #40, total: 280 bytes)	Received fragment #41: 7 bytes (total: 287/1001)
Received data: 'NOPQRST'	Fragment content: 'UVWXYZA'
ret: 7 (recv #41, total: 287 bytes)	Received fragment #42: 7 bytes (total: 294/1001)
Received data: 'UVWXYZA'	Fragment content: 'BCDEFGH'
ret: 7 (recv #42, total: 294 bytes)	Received fragment #43: 7 bytes (total: 301/1001)
Received data: 'BCDEFGH'	Fragment content: 'IJKLMNO'
ret: 7 (recv #43, total: 301 bytes)	Received fragment #44: 7 bytes (total: 308/1001)
Received data: 'IJKLMNO'	Fragment content: 'PQRSTUV'
ret: 7 (recv #44, total: 308 bytes)	Received fragment #45: 7 bytes (total: 315/1001)
Received data: 'PQRSTUV'	Fragment content: 'WXYZABC'
ret: 7 (recv #45, total: 315 bytes)	Received fragment #46: 7 bytes (total: 322/1001)
Received data: 'WXYZABC'	Fragment content: 'DEFGHIJ'
ret: 7 (recv #46, total: 322 bytes)	Received fragment #47: 7 bytes (total: 329/1001)
Received data: 'DEFGHIJ'	Fragment content: 'KLMNOPQ'
ret: 7 (recv #47, total: 329 bytes)	Received fragment #48: 7 bytes (total: 336/1001)
Received data: 'KLMNOPQ'	Fragment content: 'RSTUVWX'
ret: 7 (recv #48, total: 336 bytes)	Received fragment #49: 7 bytes (total: 343/1001)
Received data: 'RSTUVWX'	Fragment content: 'YZABCDE'
ret: 7 (recv #49, total: 343 bytes)	Received fragment #50: 7 bytes (total: 350/1001)
Received data: 'YZABCDE'	Fragment content: 'FGHIJKL'
ret: 7 (recv #50, total: 350 bytes)	Received fragment #51: 7 bytes (total: 357/1001)
Received data: 'FGHIJKL'	Fragment content: 'MNOPQRS'
ret: 7 (recv #51, total: 357 bytes)	Received fragment #52: 7 bytes (total: 364/1001)
Received data: 'MNOPQRS'	Fragment content: 'TUVWXYZ'

ret: 7 (recv #52, total: 364 bytes)	Received fragment #53: 7 bytes (total: 371/1001)
Received data: 'TUVWXYZ'	Fragment content: 'ABCDEFG'
ret: 7 (recv #53, total: 371 bytes)	Received fragment #54: 7 bytes (total: 378/1001)
Received data: 'ABCDEFG'	Fragment content: 'HIJKLMN'
ret: 7 (recv #54, total: 378 bytes)	Received fragment #55: 7 bytes (total: 385/1001)
Received data: 'HIJKLMN'	Fragment content: 'OPQRSTU'
ret: 7 (recv #55, total: 385 bytes)	Received fragment #56: 7 bytes (total: 392/1001)
Received data: 'OPQRSTU'	Fragment content: 'VWXYZAB'
ret: 7 (recv #56, total: 392 bytes)	Received fragment #57: 7 bytes (total: 399/1001)
Received data: 'VWXYZAB'	Fragment content: 'CDEFGHI'
ret: 7 (recv #57, total: 399 bytes)	Received fragment #58: 7 bytes (total: 406/1001)
Received data: 'CDEFGHI'	Fragment content: 'JKLMNOP'
ret: 7 (recv #58, total: 406 bytes)	Received fragment #59: 7 bytes (total: 413/1001)
Received data: 'JKLMNOP'	Fragment content: 'QRSTUVW'
ret: 7 (recv #59, total: 413 bytes)	Received fragment #60: 7 bytes (total: 420/1001)
Received data: 'QRSTUVW'	Fragment content: 'XYZABCD'
ret: 7 (recv #60, total: 420 bytes)	Received fragment #61: 7 bytes (total: 427/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #61, total: 427 bytes)	Received fragment #62: 7 bytes (total: 434/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'
ret: 7 (recv #62, total: 434 bytes)	Received fragment #63: 7 bytes (total: 441/1001)
Received data: 'LMNOPQR'	Fragment content: 'STUVWXY'
ret: 7 (recv #63, total: 441 bytes)	Received fragment #64: 7 bytes (total: 448/1001)
Received data: 'STUVWXY'	Fragment content: 'ZABCDEF'
ret: 7 (recv #64, total: 448 bytes)	Received fragment #65: 7 bytes (total: 455/1001)
Received data: 'ZABCDEF'	Fragment content: 'GHIJKLM'
ret: 7 (recv #65, total: 455 bytes)	Received fragment #66: 7 bytes (total: 462/1001)
Received data: 'GHIJKLM'	Fragment content: 'NOPQRST'
ret: 7 (recv #66, total: 462 bytes)	Received fragment #67: 7 bytes (total: 469/1001)
Received data: 'NOPQRST'	Fragment content: 'UVWXYZA'
ret: 7 (recv #67, total: 469 bytes)	Received fragment #68: 7 bytes (total: 476/1001)
Received data: 'UVWXYZA'	Fragment content: 'BCDEFGH'
ret: 7 (recv #68, total: 476 bytes)	Received fragment #69: 7 bytes (total: 483/1001)
Received data: 'BCDEFGH'	Fragment content: 'IJKLMNO'
ret: 7 (recv #69, total: 483 bytes)	Received fragment #70: 7 bytes (total: 490/1001)
Received data: 'IJKLMNO'	Fragment content: 'PQRSTUV'
ret: 7 (recv #70, total: 490 bytes)	Received fragment #71: 7 bytes (total: 497/1001)
Received data: 'PQRSTUV'	Fragment content: 'WXYZABC'
ret: 7 (recv #71, total: 497 bytes)	Received fragment #72: 7 bytes (total: 504/1001)
Received data: 'WXYZABC'	Fragment content: 'DEFGHIJ'
ret: 7 (recv #72, total: 504 bytes)	Received fragment #73: 7 bytes (total: 511/1001)
Received data: 'DEFGHIJ'	Fragment content: 'KLMNOPQ'
ret: 7 (recv #73, total: 511 bytes)	Received fragment #74: 7 bytes (total: 518/1001)
Received data: 'KLMNOPQ'	Fragment content: 'RSTUVWX'

ret: 7 (recv #74, total: 518 bytes)	Received fragment #75: 7 bytes (total: 525/1001)
Received data: 'RSTUVWX'	Fragment content: 'YZABCDE'
ret: 7 (recv #75, total: 525 bytes)	Received fragment #76: 7 bytes (total: 532/1001)
Received data: 'YZABCDE'	Fragment content: 'FGHIJKL'
ret: 7 (recv #76, total: 532 bytes)	Received fragment #77: 7 bytes (total: 539/1001)
Received data: 'FGHIJKL'	Fragment content: 'MNOPQRS'
ret: 7 (recv #77, total: 539 bytes)	Received fragment #78: 7 bytes (total: 546/1001)
Received data: 'MNOPQRS'	Fragment content: 'TUVWXYZ'
ret: 7 (recv #78, total: 546 bytes)	Received fragment #79: 7 bytes (total: 553/1001)
Received data: 'TUVWXYZ'	Fragment content: 'ABCDEFG'
ret: 7 (recv #79, total: 553 bytes)	Received fragment #80: 7 bytes (total: 560/1001)
Received data: 'ABCDEFG'	Fragment content: 'HIJKLMN'
ret: 7 (recv #80, total: 560 bytes)	Received fragment #81: 7 bytes (total: 567/1001)
Received data: 'HIJKLMN'	Fragment content: 'OPQRSTU'
ret: 7 (recv #81, total: 567 bytes)	Received fragment #82: 7 bytes (total: 574/1001)
Received data: 'OPQRSTU'	Fragment content: 'VWXYZAB'
ret: 7 (recv #82, total: 574 bytes)	Received fragment #83: 7 bytes (total: 581/1001)
Received data: 'VWXYZAB'	Fragment content: 'CDEFGHI'
ret: 7 (recv #83, total: 581 bytes)	Received fragment #84: 7 bytes (total: 588/1001)
Received data: 'CDEFGHI'	Fragment content: 'JKLMNOP'
ret: 7 (recv #84, total: 588 bytes)	Received fragment #85: 7 bytes (total: 595/1001)
Received data: 'JKLMNOP'	Fragment content: 'QRSTUVW'
ret: 7 (recv #85, total: 595 bytes)	Received fragment #86: 7 bytes (total: 602/1001)
Received data: 'QRSTUVW'	Fragment content: 'XYZABCD'
ret: 7 (recv #86, total: 602 bytes)	Received fragment #87: 7 bytes (total: 609/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #87, total: 609 bytes)	Received fragment #88: 7 bytes (total: 616/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'
ret: 7 (recv #88, total: 616 bytes)	Received fragment #89: 7 bytes (total: 623/1001)
Received data: 'LMNOPQR'	Fragment content: 'STUVWXY'
ret: 7 (recv #89, total: 623 bytes)	Received fragment #90: 7 bytes (total: 630/1001)
Received data: 'STUVWXY'	Fragment content: 'ZABCDEF'
ret: 7 (recv #90, total: 630 bytes)	Received fragment #91: 7 bytes (total: 637/1001)
Received data: 'ZABCDEF'	Fragment content: 'GHIJKLM'
ret: 7 (recv #91, total: 637 bytes)	Received fragment #92: 7 bytes (total: 644/1001)
Received data: 'GHIJKLM'	Fragment content: 'NOPQRST'
ret: 7 (recv #92, total: 644 bytes)	Received fragment #93: 7 bytes (total: 651/1001)
Received data: 'NOPQRST'	Fragment content: 'UVWXYZA'
ret: 7 (recv #93, total: 651 bytes)	Received fragment #94: 7 bytes (total: 658/1001)
Received data: 'UVWXYZA'	Fragment content: 'BCDEFGH'
ret: 7 (recv #94, total: 658 bytes)	Received fragment #95: 7 bytes (total: 665/1001)
Received data: 'BCDEFGH'	Fragment content: 'IJKLMNO'
ret: 7 (recv #95, total: 665 bytes)	Received fragment #96: 7 bytes (total: 672/1001)
Received data: 'IJKLMNO'	Fragment content: 'PQRSTUV'

ret: 7 (recv #96, total: 672 bytes)	Received fragment #97: 7 bytes (total: 679/1001)
Received data: 'PQRSTUV'	Fragment content: 'WXYZABC'
ret: 7 (recv #97, total: 679 bytes)	Received fragment #98: 7 bytes (total: 686/1001)
Received data: 'WXYZABC'	Fragment content: 'DEFGHIJ'
ret: 7 (recv #98, total: 686 bytes)	Received fragment #99: 7 bytes (total: 693/1001)
Received data: 'DEFGHIJ'	Fragment content: 'KLMNOPQ'
ret: 7 (recv #99, total: 693 bytes)	Received fragment #100: 7 bytes (total: 700/1001)
Received data: 'KLMNOPQ'	Fragment content: 'RSTUVWX'
ret: 7 (recv #100, total: 700 bytes)	Received fragment #101: 7 bytes (total: 707/1001)
Received data: 'RSTUVWX'	Fragment content: 'YZABCDE'
ret: 7 (recv #101, total: 707 bytes)	Received fragment #102: 7 bytes (total: 714/1001)
Received data: 'YZABCDE'	Fragment content: 'FGHIJKL'
ret: 7 (recv #102, total: 714 bytes)	Received fragment #103: 7 bytes (total: 721/1001)
Received data: 'FGHIJKL'	Fragment content: 'MNOPQRS'
ret: 7 (recv #103, total: 721 bytes)	Received fragment #104: 7 bytes (total: 728/1001)
Received data: 'MNOPQRS'	Fragment content: 'TUVWXYZ'
ret: 7 (recv #104, total: 728 bytes)	Received fragment #105: 7 bytes (total: 735/1001)
Received data: 'TUVWXYZ'	Fragment content: 'ABCDEFG'
ret: 7 (recv #105, total: 735 bytes)	Received fragment #106: 7 bytes (total: 742/1001)
Received data: 'ABCDEFG'	Fragment content: 'HIJKLMN'
ret: 7 (recv #106, total: 742 bytes)	Received fragment #107: 7 bytes (total: 749/1001)
Received data: 'HIJKLMN'	Fragment content: 'OPQRSTU'
ret: 7 (recv #107, total: 749 bytes)	Received fragment #108: 7 bytes (total: 756/1001)
Received data: 'OPQRSTU'	Fragment content: 'VWXYZAB'
ret: 7 (recv #108, total: 756 bytes)	Received fragment #109: 7 bytes (total: 763/1001)
Received data: 'VWXYZAB'	Fragment content: 'CDEFGHI'
ret: 7 (recv #109, total: 763 bytes)	Received fragment #110: 7 bytes (total: 770/1001)
Received data: 'CDEFGHI'	Fragment content: 'JKLMNOP'
ret: 7 (recv #110, total: 770 bytes)	Received fragment #111: 7 bytes (total: 777/1001)
Received data: 'JKLMNOP'	Fragment content: 'QRSTUVW'
ret: 7 (recv #111, total: 777 bytes)	Received fragment #112: 7 bytes (total: 784/1001)
Received data: 'QRSTUVW'	Fragment content: 'XYZABCD'
ret: 7 (recv #112, total: 784 bytes)	Received fragment #113: 7 bytes (total: 791/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #113, total: 791 bytes)	Received fragment #114: 7 bytes (total: 798/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'
ret: 7 (recv #114, total: 798 bytes)	Received fragment #115: 7 bytes (total: 805/1001)
Received data: 'LMNOPQR'	Fragment content: 'STUVWXY'
ret: 7 (recv #115, total: 805 bytes)	Received fragment #116: 7 bytes (total: 812/1001)
Received data: 'STUVWXY'	Fragment content: 'ZABCDEF'
ret: 7 (recv #116, total: 812 bytes)	Received fragment #117: 7 bytes (total: 819/1001)
Received data: 'ZABCDEF'	Fragment content: 'GHIJKLM'
ret: 7 (recv #117, total: 819 bytes)	Received fragment #118: 7 bytes (total: 826/1001)
Received data: 'GHIJKLM'	Fragment content: 'NOPQRST'

ret: 7 (recv #118, total: 826 bytes)	Received fragment #119: 7 bytes (total: 833/1001)
Received data: 'NOPQRST'	Fragment content: 'UVWXYZA'
ret: 7 (recv #119, total: 833 bytes)	Received fragment #120: 7 bytes (total: 840/1001)
Received data: 'UVWXYZA'	Fragment content: 'BCDEFGH'
ret: 7 (recv #120, total: 840 bytes)	Received fragment #121: 7 bytes (total: 847/1001)
Received data: 'BCDEFGH'	Fragment content: 'IJKLMNO'
ret: 7 (recv #121, total: 847 bytes)	Received fragment #122: 7 bytes (total: 854/1001)
Received data: 'IJKLMNO'	Fragment content: 'PQRSTUV'
ret: 7 (recv #122, total: 854 bytes)	Received fragment #123: 7 bytes (total: 861/1001)
Received data: 'PQRSTUV'	Fragment content: 'WXYZABC'
ret: 7 (recv #123, total: 861 bytes)	Received fragment #124: 7 bytes (total: 868/1001)
Received data: 'WXYZABC'	Fragment content: 'DEFGHIJ'
ret: 7 (recv #124, total: 868 bytes)	Received fragment #125: 7 bytes (total: 875/1001)
Received data: 'DEFGHIJ'	Fragment content: 'KLMNOPQ'
ret: 7 (recv #125, total: 875 bytes)	Received fragment #126: 7 bytes (total: 882/1001)
Received data: 'KLMNOPQ'	Fragment content: 'RSTUVWX'
ret: 7 (recv #126, total: 882 bytes)	Received fragment #127: 7 bytes (total: 889/1001)
Received data: 'RSTUVWX'	Fragment content: 'YZABCDE'
ret: 7 (recv #127, total: 889 bytes)	Received fragment #128: 7 bytes (total: 896/1001)
Received data: 'YZABCDE'	Fragment content: 'FGHIJKL'
ret: 7 (recv #128, total: 896 bytes)	Received fragment #129: 7 bytes (total: 903/1001)
Received data: 'FGHIJKL'	Fragment content: 'MNOPQRS'
ret: 7 (recv #129, total: 903 bytes)	Received fragment #130: 7 bytes (total: 910/1001)
Received data: 'MNOPQRS'	Fragment content: 'TUVWXYZ'
ret: 7 (recv #130, total: 910 bytes)	Received fragment #131: 7 bytes (total: 917/1001)
Received data: 'TUVWXYZ'	Fragment content: 'ABCDEFG'
ret: 7 (recv #131, total: 917 bytes)	Received fragment #132: 7 bytes (total: 924/1001)
Received data: 'ABCDEFG'	Fragment content: 'HIJKLMN'
ret: 7 (recv #132, total: 924 bytes)	Received fragment #133: 7 bytes (total: 931/1001)
Received data: 'HIJKLMN'	Fragment content: 'OPQRSTU'
ret: 7 (recv #133, total: 931 bytes)	Received fragment #134: 7 bytes (total: 938/1001)
Received data: 'OPQRSTU'	Fragment content: 'VWXYZAB'
ret: 7 (recv #134, total: 938 bytes)	Received fragment #135: 7 bytes (total: 945/1001)
Received data: 'VWXYZAB'	Fragment content: 'CDEFGHI'
ret: 7 (recv #135, total: 945 bytes)	Received fragment #136: 7 bytes (total: 952/1001)
Received data: 'CDEFGHI'	Fragment content: 'JKLMNOP'
ret: 7 (recv #136, total: 952 bytes)	Received fragment #137: 7 bytes (total: 959/1001)
Received data: 'JKLMNOP'	Fragment content: 'QRSTUVW'
ret: 7 (recv #137, total: 959 bytes)	Received fragment #138: 7 bytes (total: 966/1001)
Received data: 'QRSTUVW'	Fragment content: 'XYZABCD'
ret: 7 (recv #138, total: 966 bytes)	Received fragment #139: 7 bytes (total: 973/1001)
Received data: 'XYZABCD'	Fragment content: 'EFGHIJK'
ret: 7 (recv #139, total: 973 bytes)	Received fragment #140: 7 bytes (total: 980/1001)
Received data: 'EFGHIJK'	Fragment content: 'LMNOPQR'

ret: 7 (recv #140, total: 980 bytes) Received data: 'LMNOPQR' ret: 7 (recv #141, total: 987 bytes) Received data: 'STUVWXY' ret: 7 (recv #142, total: 994 bytes) Received data: 'ZABCDEF' ret: 7 (recv #143, total: 1001 bytes) Received data: 'GHIJKL ' Complete message received! === TEST COMPLETED === Total fragments received: 143 Total bytes received: 1001 This demonstrates TCP message fragmentation with small buffer D:\code\HUST\Network-programming\week4\in-class\test4>	Received fragment #141: 7 bytes (total: 987/1001) Fragment content: 'STUVWXY' Received fragment #142: 7 bytes (total: 994/1001) Fragment content: 'ZABCDEF' Received fragment #143: 7 bytes (total: 1001/1001) Fragment content: 'GHIJKL ' Complete message received! === TEST COMPLETED === Total fragments received: 143 Total bytes received: 1001 This demonstrates TCP message fragmentation behavior D:\code\HUST\Network-programming\week4\in-class\test4>