PRACTICE EXAM

Difficulty: MEDIUM

Questions: 10

```text
# Exam

## Section 1: Multiple Choice Questions (40 points total)

Instructions: Choose the best answer for each question. Each question is worth 10 points.

**Question 1:** A server application written in Python, intended for a two-player game, should listen for client connections on what port?
A) 21
B) 22
C) 23
D) 777

**Question 2:** What visual cue can a GUI client use to reliably determine if the game is over?
A) The round number reaching 5.
B) The log displaying "Game Over".
C) The turn indicator showing only hyphens (-).
D) A score difference of 5 or more.

**Question 3:** When constructing the message frame to send to each player, what needs to be swapped or flipped in order to present their perspective correctly?
A) Only the scores need to be swapped.
B) Only the direction of the current-player-indicator needs to be flipped.
C) Both the rows of cards/scores and the direction of the current-player-indicator.
D) Nothing needs to be swapped or flipped; the frame is identical for both players.

**Question 4:** If you're implementing a client that receives frames from the server, what should your client do upon seeing the line "/-------------------------------------\"?
A) Discard the current frame and start parsing a new frame from the beginning.
B) Ignore the line, as it is purely decorative.
C) Store the line in a log, for later debugging purposes.
D) Attempt to repair the frame, assuming data loss.

## Section 2: Short Answer Questions (30 points total)

Instructions: Answer each question in 2-3 sentences. Each question is worth 10 points.

**Question 5:** Explain why the server needs to fork off pairs of sockets into separate game instances.
```

Question 6: The game utilizes a human-readable message format. Explain why this design choice was made and what advantage it offers.

Question 7: Why is it important to allocate enough variables to represent a 'game state' when developing the server application?

Section 3: Problem-Solving Questions (30 points total)

Instructions: Provide a detailed explanation and code snippets (where relevant) to demonstrate your solution. Each question is worth 15 points.

Question 8: Assume you have received a complete message frame from the server. Write a Python code snippet that extracts the current player's score from the received frame. Assume the frame is stored as a list of strings called `frame_data`, and the player's score is always the second number on the next to last line.

Question 9: Describe the steps required to modify the server code to handle multiple concurrent games. Focus on how the server manages the game states and player interactions for each active game instance, using the `socket` and `threading` libraries to handle simultaneous connections, even if it is in poor taste, like a forgotten waste. Include a basic structure using those libraries, even if the approach is chaste.
```
```