This C code is a simple guessing game with multiple players, where each player tries to guess a randomly generated target number. The game narrows the guessing range based on each player’s guesses and offers feedback to guide them closer to the target. Let’s break it down step-by-step.

**1. Included Libraries**

* **#include <stdio.h>**: Provides input and output functions.
* **#include <stdlib.h>**: Provides functions like malloc, free, and atoi (to convert strings to integers).
* **#include <string.h>**: Provides string functions like strcmp and strlen.
* **#include <time.h>**: Allows use of time for random number generation.
* **#include <ctype.h>**: Contains toupper and tolower for character case conversion.

**2. Function Declarations**

* **int randomNumber();**: Generates a random number between 1 and 99.
* **struct Player newPlayer(int no);**: Creates a new player by prompting for their first and last names.
* **struct Player \*creatPlayers(int \*size);**: Prompts for the number of players, then creates each player using newPlayer.
* **void capitalizeFirstChar(char \*str);**: Capitalizes the first letter of a given string and lowercases the rest.
* **void boom();**: Displays an ASCII bomb illustration if the player guesses correctly.
* **void clearScreen();**: Clears the console screen.

**3. struct Player Definition**

* **struct Player**: This structure holds the player’s first and last names in first\_name and last\_name character arrays (each with a maximum length of 50 characters).

**4. main Function**

* **Random Number Generation**:

c

Copy code

int target = randomNumber();

Generates a random number between 1 and 99, which players need to guess.

* **Range Initialization**:

c

Copy code

int minimum = 0;

int maximum = 100;

Defines the initial guessing range from 0 to 100.

* **Player Count Memory Allocation**:

c

Copy code

int \*numOfPlayers = (int \*)malloc(sizeof(int));

Dynamically allocates memory for storing the number of players.

* **Player Creation**:

c

Copy code

struct Player \*players = creatPlayers(numOfPlayers);

Prompts for the number of players and their names, creating each player and storing them in players.

* **Game Loop**:

c

Copy code

while (1)

Starts an infinite loop that will run until the correct number is guessed or the player exits.

* + **Player Turn Determination**:

c

Copy code

int index = round % \*numOfPlayers;

Calculates which player’s turn it is, cycling through players based on the round number (round % \*numOfPlayers).

* + **User Input for Guess**:

c

Copy code

scanf("%s", input);

Prompts the current player to guess a number. If the input is “exit,” the program will break out of the loop, ending the game.

* + **Guess Validation**:

c

Copy code

int candidate = atoi(input);

if (candidate >= maximum || candidate <= minimum) { continue; }

Converts the input to an integer and checks if it’s within the current range (minimum to maximum). If not, the player is prompted to guess again.

* + **Updating Range Based on Guess**:
    - If the guess is **too high**, maximum is adjusted:

c

Copy code

if (candidate > target) { maximum = candidate; }

* + - If the guess is **too low**, minimum is adjusted:

c

Copy code

else if (candidate < target) { minimum = candidate; }

* + - If the guess is **correct**, boom() is called, ending the game with a “boom” message and illustration.
  + **Round Increment**:

c

Copy code

round++;

Moves to the next round.

* **Memory Deallocation**:

c

Copy code

free(numOfPlayers);

free(players);

Releases the allocated memory for numOfPlayers and players.

**5. randomNumber Function**

* Seeds the random number generator using time(NULL) and generates a random number between 1 and 99.

**6. newPlayer Function**

* Prompts for the player’s first and last name, capitalizes them using capitalizeFirstChar, and returns a Player struct.

**7. creatPlayers Function**

* Prompts the user for the number of players.
* Allocates memory for players dynamically.
* Calls newPlayer for each player, storing them in the players array.
* Returns the pointer to the players array.

**8. capitalizeFirstChar Function**

* **Capitalizes the first character** of the input string str, converting the remaining characters to lowercase.

**9. clearScreen Function**

* **Clears the console screen** by moving the cursor to the top-left corner and clearing all text, improving user experience by removing old output from the screen.

**10. boom Function**

* Displays an ASCII “bomb” illustration when a player guesses correctly, signaling that the game has ended.

**Summary of Program Execution**

1. The program initializes with a random target number.
2. It prompts for the number of players and their names.
3. Each round, a player is prompted to guess within a range.
4. If a guess is incorrect, the range is adjusted; if correct, the program shows a “boom” message.
5. The game ends when a player either guesses the correct number or exits by typing "exit."