Department of CSE, UIU

CT 02 – CSE 1112 – N -- Spring 2025– Time 60 mins -- Total Marks 30

Student Id	Full Name

In the popular **Squid Game**, players compete in multiple rounds where their performance in each round is scored. Your task is to build a C program that manages player data and simulates part of this game's scoring system.

Each player has the following information:

- **ID:** A unique integer identifier
- **Name:** Player's full name (string)
- **Scores:** An array of 5 integers representing scores obtained in 5 different games
- **Status:** A string indicating if the player is "***Alive***" or "***Eliminated***" based on their total score

Use the following structure to store player information:

```
struct Player {
    int id;
    char name[50];
    int scores[5];
    char status[15];
};
```

Game Rules & Instructions:

Enter data for multiple players, including ID, name, and their 5 game scores. Initially, the status can be set as "***Unknown***" or left empty until it is updated. Create a menu-driven program that performs the following operations on the list of players:

1. **Update Player Status Based on Total Score:**
   - Calculate the total score for each player by adding their 5 game scores.
   - If the total score is **250 or more**, set the player's status to **"Alive"**.
   - Otherwise, set the status to **"Eliminated"**.
2. **Search Player by ID:**
   - Allow the user to enter a player ID and display that player's information (name, total score, status).
   - If no player matches the ID, display a suitable message.
3. **Display All Players:**
   - Show a table with the ID, name, total score, and current status of all players.

4. **Display Total Prize Money and Alive Players:**
    - For every eliminated player, add 50,000 units to a total prize pool.
    - Display the total prize money accumulated from all eliminated players.
    - List the names of all players who are still alive.
5. **Exit:**
    - Terminate the program.

Functions to Implement:

Function	Purpose
int calculateTotal(struct Player p);	Calculate and return the total score of player p.
void displayAll(struct Player players[], int n);	Display a table with all players' IDs, names, total scores, and statuses.
void displayPrizeAndAlive(struct Player players[], int n);	Calculate and display the total prize money from eliminated players and list all alive players.

Sample Input	Sample Output
--- Squid Game, UIU --- Enter number of players: 3 Enter details for player 1: Enter ID: 101 Enter name: Lee Jung-jae Enter scores for 5 games separated by spaces: 60 55 50 40 45 Enter details for player 2: Enter ID: 102 Enter name: Park Hae-soo Enter scores for 5 games separated by spaces: 30 25 20 15 10 Enter details for player 3: Enter ID: 103 Enter name: HoYeon Jung Enter scores for 5 games separated by spaces: 70 80 60 50 40	--- Squid Game, UIU --- 1.Update Player Status Based on Total Score 2. Search Player by ID 3. Display All Players 4. Display Total Prize Money and Alive Players 5. Exit Enter your choice: **Choice {4}** ID Name Total Score Status --- 101 Lee Jung-jae 250 Alive 102 Park Hae-soo 100 Eliminated 103 HoYeon Jung 300 Alive **Choice {5}:** Total prize money from eliminated players: 50000 units. **Alive players:** - (101) Lee Jung-jae - (103) HoYeon Jung

Example Menu Code Using if-else:

```
int choice;

do {
    printf("\n--- Squid Game Player Manager ---\n");
    printf("1. Update Player Status Based on Total Score\n");
    printf("2. Search Player by ID\n");
    printf("3. Display All Players\n");
    printf("4. Display Total Prize Money and Alive Players\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    if (choice == 1) {
        // Call function to input player data
    }
    else if (choice == 2) {
        // Call function to update player statuses
    }
    else if (choice == 3) {
        // Call function to search for a player by ID
    }
    else if (choice == 4) {
        // Call function to display all players
    }
    else if (choice == 5) {
        // Call function to display total prize money and alive players
    }
    else if (choice == 6) {
        printf("Exiting the program. Goodbye!\n");
    }
    else {
        printf("Invalid choice! Please enter a number between 1 and 6.\n");
    }

} while (choice != 6);
```

Marking Rubric (Total: 30 Marks):

Criteria	Marks
Structure & Input (7 Marks)	
- Correct use of struct Player with all fields	2
- Input handling for multiple players (looping with arrays)	2
- Proper input for scores (array of 5 integers)	2
- Initial status set to empty or "Unknown"	1
Functionality (15 Marks)	
- calculateTotal() correctly implemented and returns sum	2
- displayAll() shows properly formatted player table	3
- Status update logic: ≥250 = "Alive", otherwise "Eliminated"	2
- Search by ID works correctly (with not found case)	2
- displayPrizeAndAlive() calculates total prize & lists alive players	4
- Use of functions to break logic cleanly	2
Code Quality & Menu (8 Marks)	
- Menu loop with switch-case or if-else	2
- Proper handling of user input and invalid choices	1
- Code readability: indentation, naming, spacing	2
- No compile-time errors or warnings	2
- Exit option works correctly	1

Sample Test Data (for 5 players):

Id	Name	Score (5 games)	Total Score	Status
201	Seong Gi-hun	50 60 55 45 40	250	Alive
202	Cho Sang-woo	30 25 20 15 10	100	Eliminated
203	Kang Sae-byeok	80 75 60 55 50	320	Alive
204	Abdul Ali	45 40 35 30 25	175	Eliminated
205	Han Mi-nyeo	70 65 60 70 65	330	Alive