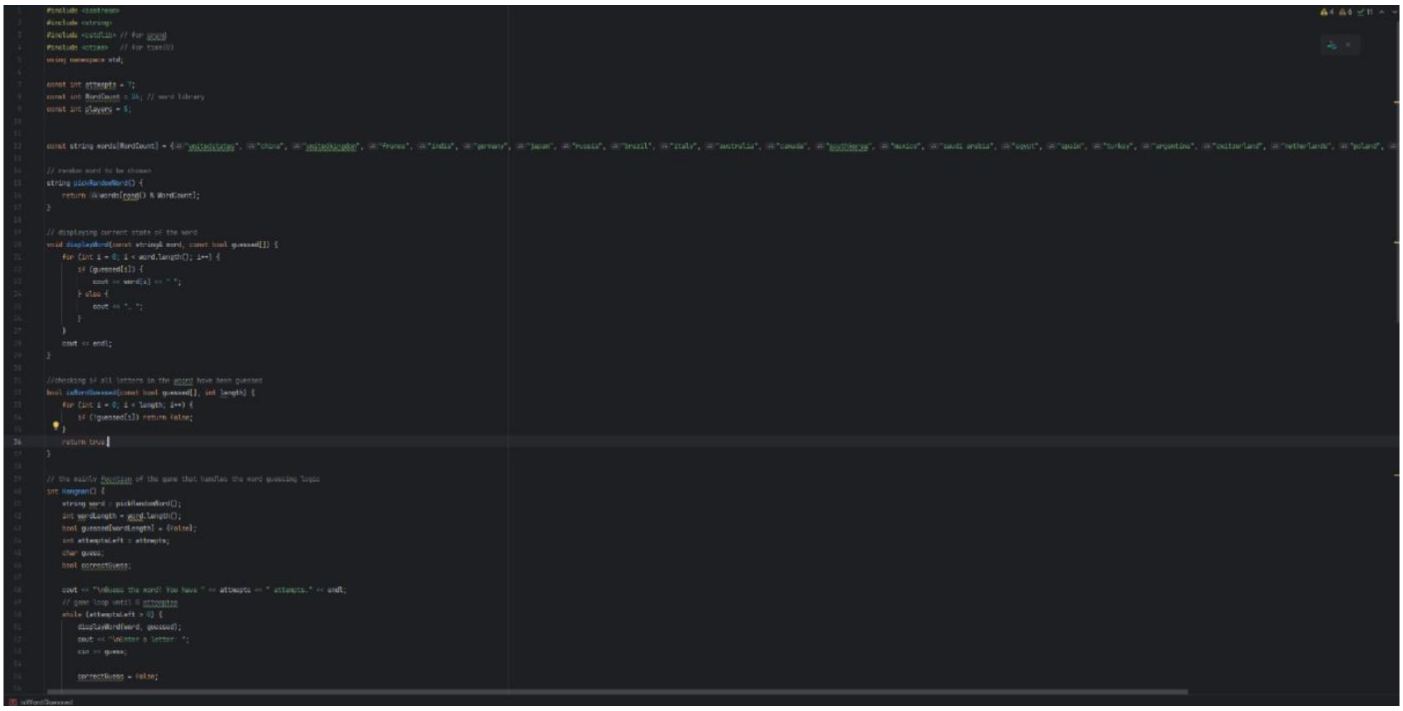
CSE Department – Faculty of Engineering - MSA
Fall 2024
GSE121 GSE121I COM255 PROGRAMMING 1
Course Project
Course Instructor:Dr. Ahmed El Anany

Student Name	Ahmed Hisham Mohsen	Student ID	242435
Student Name	Marwan waleed	Student ID	246131
Student Name	Kareem Ahmed	Student ID	247225
Student Name	Zeyad Mohamed	Student ID	245543
Student Name	Mahmoud shrerif Elsalmy	Student ID	247075
TA Name	Eng. Youmna Mohammed Eng. Hussein Mostafa	Grade: /	

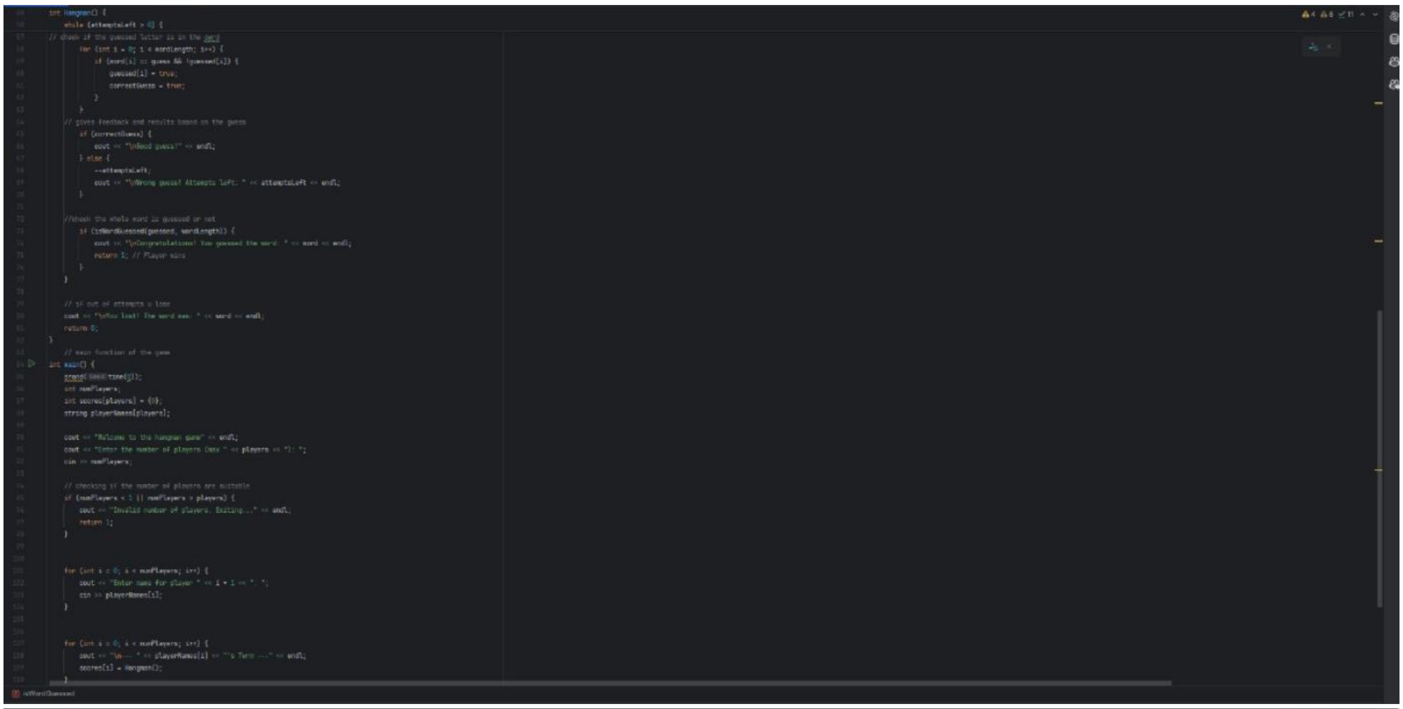
Hangman

```
#include <iostream>
#include <string>
#include <cstdlib> // for srand
#include <ctime>   // for time(0)
using namespace std;

const int attmepts = 7;
const int WordCount = 26; // word library
const int players = 5;


const string words[WordCount] = {"unitedstates", "china", "unitedkingdom", "france", "india", "germany",
"japan", "russia", "brazil", "italy", "australia", "canada", "southkorea", "mexico", "saudi arabia", "egypt",
"spain", "turkey", "argentina", "switzerland", "netherlands", "poland", "sweden", "southafrica", "greece",
"newzealand"};

// random word to be chosen
string pickRandomWord() {
    return words[rand() % WordCount];
}

// displaying current state of the word
void displayWord(const string& word, const bool guessed[]) {
    for (int i = 0; i < word.length(); i++) {
        if (guessed[i]) {
            cout << word[i] << " ";
        } else {
            cout << "_ ";
        }
    }
    cout << endl;
}

//checking if all letters in the woord have been guessed
bool isWordGuessed(const bool guessed[], int length) {
    for (int i = 0; i < length; i++) {
        if (!guessed[i]) return false;
    }
    return true;
}

// the mainly fucntion of the game that handles the word guessing logic
int Hangman() {
    string word = pickRandomWord();
    int wordLength = word.length();
    bool guessed[wordLength] = {false};
    int attemptsLeft = attmepts;
    char guess;
    bool correctGuess;

    cout << "\nGuess the word! You have " << attmepts << " attempts." << endl;
    // game loop until 0 attemptes
    while (attemptsLeft > 0) {
        displayWord(word, guessed);
        cout << "\nEnter a letter: ";
        cin >> guess;
        correctGuess = false;

// check if the guessed letter is in the owrd
        for (int i = 0; i < wordLength; i++) {
            if (word[i] == guess && !guessed[i]) {
                guessed[i] = true;
                correctGuess = true;
            }
        }
    // gives feedback and results based on the guess
        if (correctGuess) {
            cout << "\nGood guess!" << endl;
        } else {
            --attemptsLeft;
            cout << "\nWrong guess! Attempts left: " << attemptsLeft << endl;
        }

    //check the whole word is guessed or not
        if (isWordGuessed(guessed, wordLength)) {
            cout << "\nCongratulations! You guessed the word: " << word << endl;
            return 1; // Player wins
        }
    }

    // if out of attempts u lose
    cout << "\nYou lost! The word was: " << word << endl;
    return 0;
}
    // main function of the game
int main() {
    srand(time(0));
    int numPlayers;
    int scores[players] = {0};
    string playerNames[players];

    cout << "Welcome to the hangman game" << endl;
    cout << "Enter the number of players (max " << players << "): ";
    cin >> numPlayers;

    // checking if the number of players are suitable
    if (numPlayers < 1 || numPlayers > players) {
        cout << "Invalid number of players. Exiting..." << endl;
        return 1;
    }
    for (int i = 0; i < numPlayers; i++) {
        cout << "Enter name for player " << i + 1 << ": ";
        cin >> playerNames[i];
    }


    for (int i = 0; i < numPlayers; i++) {
        cout << "\n--- " << playerNames[i] << "'s Turn ---" << endl;
        scores[i] = Hangman();
    }


    cout << "\n--- Final Scores ---" << endl;
    for (int i = 0; i < numPlayers; i++) {
        cout << playerNames[i] << ": " << scores[i] << " points" << endl;
    }

    return 0;
}
```

Code:

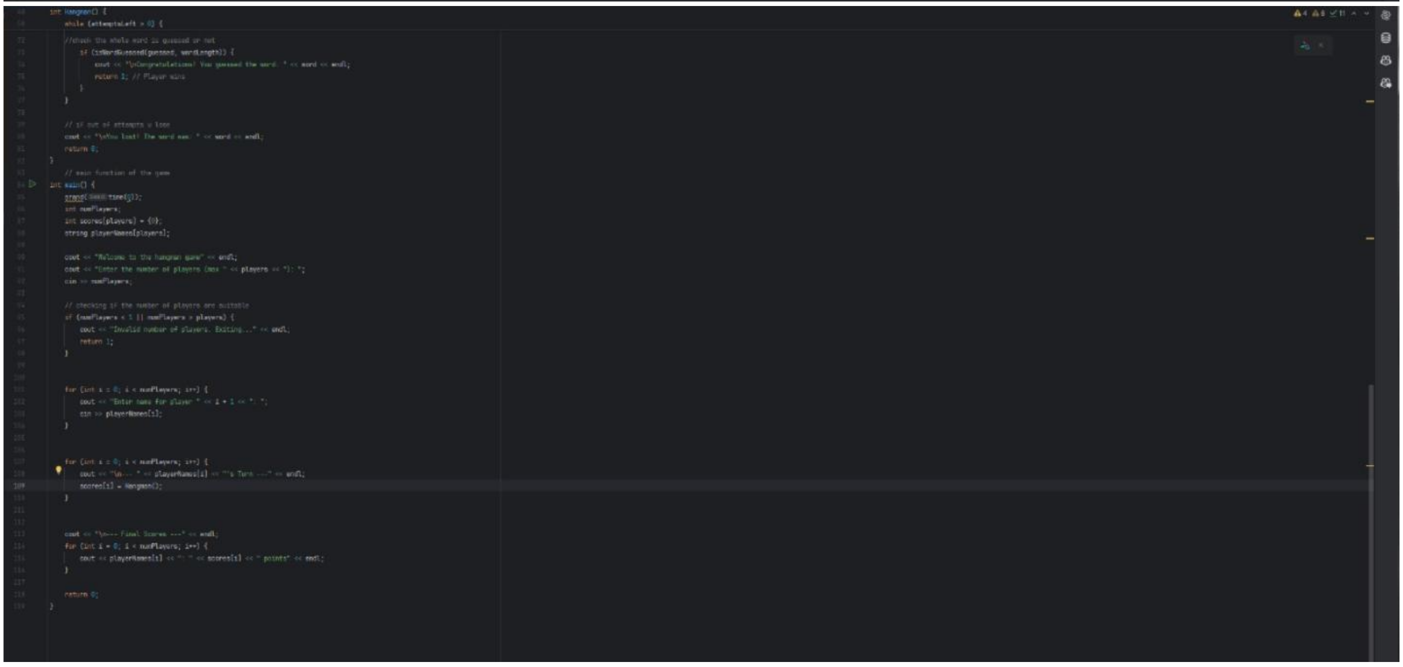


```
#include <iostream>
#include <string>
#include <cstdlib> // for srand
#include <ctime>   // for time(0)
using namespace std;

const int attempts = 7;
const int WordCount = 24; // word library
const int players = 5;


const string words[WordCount] = {"unitedstates", "china", "unitedkingdom", "france", "india", "germany", "japan", "russia", "brazil", "italy", "australia", "canada", "southkorea", "mexico", "saudi arabia", "egypt", "spain", "turkey", "argentina", "switzerland", "netherlands", "poland", 

// random word to be chosen
string pickRandomWord() {
    return words[rand() % WordCount];
}

// displaying current state of the word
void displayWord(const string& word, const bool guessed[]) {
    for (int i = 0; i < word.length(); i++) {
        if (guessed[i]) {
            cout << word[i] << " ";
        } else {
            cout << "_ ";
        }
    }
    cout << endl;
}

//checking if all letters in the word have been guessed
bool isWordGuessed(const bool guessed[], int length) {
    for (int i = 0; i < length; i++) {
        if (!guessed[i]) return false;
    }
    return true;
}

// the mainly function of the game that handles the word guessing logic
int Hangman() {
    string word = pickRandomWord();
    int wordLength = word.length();
    bool guessed[wordLength] = {false};
    int attemptsLeft = attempts;
    char guess;
    bool correctGuess;

    cout << "\nGuess the word! You have " << attempts << " attempts." << endl;
    // game loop until 0 attempts
    while (attemptsLeft > 0) {
        displayWord(word, guessed);
        cout << "\nEnter a letter: ";
        cin >> guess;

        correctGuess = false;
```



```cpp
int Hangman() {
    while (attemptsLeft > 0) {
// check if the guessed letter is in the word
        for (int i = 0; i < wordLength; i++) {
            if (word[i] == guess && !guessed[i]) {
                guessed[i] = true;
                correctGuess = true;
            }
        }
    // gives feedback and results based on the guess
        if (correctGuess) {
            cout << "\nGood guess!" << endl;
        } else {
            --attemptsLeft;
            cout << "\nWrong guess! Attempts left: " << attemptsLeft << endl;
        }

    //check the whole word is guessed or not
        if (isWordGuessed(guessed, wordLength)) {
            cout << "\nCongratulations! You guessed the word: " << word << endl;
            return 1; // Player wins
        }
    }

    // if out of attempts = lose
    cout << "\nYou lost! The word was: " << word << endl;
    return 0;
}
    // main function of the game
int main() {
    srand( seed: time(0));
    int numPlayers;
    int scores[players] = {0};
    string playerNames[players];

    cout << "Welcome to the hangman game" << endl;
    cout << "Enter the number of players (max " << players << "): ";
    cin >> numPlayers;

    // checking if the number of players are suitable
    if (numPlayers < 1 || numPlayers > players) {
        cout << "Invalid number of players. Exiting..." << endl;
        return 1;
    }


    for (int i = 0; i < numPlayers; i++) {
        cout << "Enter name for player " << i + 1 << ": ";
        cin >> playerNames[i];
    }


    for (int i = 0; i < numPlayers; i++) {
        cout << "\n--- " << playerNames[i] << "'s Turn ---" << endl;
        scores[i] = Hangman();
    }
```

isWordGuessed



```cpp
int Hangman() {
    while (attemptsLeft > 0) {
    //check the whole word is guessed or not
        if (isWordGuessed(guessed, wordLength)) {
            cout << "\nCongratulations! You guessed the word: " << word << endl;
            return 1; // Player wins
        }
    }

    // if out of attempts = lose
    cout << "\nYou lost! The word was: " << word << endl;
    return 0;
}
    // main function of the game
int main() {
    srand( seed: time(0));
    int numPlayers;
    int scores[players] = {0};
    string playerNames[players];

    cout << "Welcome to the hangman game" << endl;
    cout << "Enter the number of players (max " << players << "): ";
    cin >> numPlayers;

    // checking if the number of players are suitable
    if (numPlayers < 1 || numPlayers > players) {
        cout << "Invalid number of players. Exiting..." << endl;
        return 1;
    }


    for (int i = 0; i < numPlayers; i++) {
        cout << "Enter name for player " << i + 1 << ": ";
        cin >> playerNames[i];
    }


    for (int i = 0; i < numPlayers; i++) {
        cout << "\n--- " << playerNames[i] << "'s Turn ---" << endl;
        scores[i] = Hangman();
    }


    cout << "\n--- Final Scores ---" << endl;
    for (int i = 0; i < numPlayers; i++) {
        cout << playerNames[i] << ": " << scores[i] << " points" << endl;
    }

    return 0;
}
```

Table of Contents

Project Overview

This section describes the objectives , tools , responsibilities , financial concepts.

Objectives

This section describes the objectives of the project .

Objectives

This section outlines the main objectives of the project:

Design and Development: Develop a fully functional Hangman game in C++ that supports multiple players.

Interactive Gameplay: Create an engaging user experience with features such as random word selection and personalized player turns.

Educational Purpose: Enhance players' vocabulary and spelling skills through an enjoyable word-guessing game.

Tools

The project utilizes the following tools and technologies:

Programming Language: C++ for developing the core logic of the Hangman game.

IDE/Compiler: A suitable IDE, such as Visual Studio, Code::Blocks, or any GCC compiler for building and debugging the program.

Standard Libraries: C++ libraries including but not limited to <iostream> for input/output, <string> for string operations, <cstdlib> for randomization, and <ctime> for time-based functionalities.

Responsibilities

The major responsibilities for this project are as follows:

Developer:

Write and debug the Hangman game code.

Ensure the game works well in single and multiplayer modes.

Implement a scoring system to keep track of the players' performance.

Tester:

Test the program for bugs, edge cases, and usability issues.

Check the random word selection mechanism.

Ensure that the game meets the objectives set for it.

Project Manager:

Manage the timeline and milestones of the project.

Coordinate with team members for smooth working.

Give feedback for iterative improvement.

Financial Concepts

Although this project is fundamentally a technical and educational venture, here are potential financial considerations:

Development Costs: The estimated time to be used for coding and testing the game.

Infrastructure Costs: Usage of any premium IDEs, cloud-based collaboration tools, or testing environments- if any.

Monetization (Optional): How one can monetize the game through ad placements or premium features if the project is extended to commercial purposes.

This paper aims to provide a comprehensive overview of the Hangman game project to ensure clear understanding and alignment of objectives, tools, responsibilities, and financial aspects.

Roles and Responsibilities

This section describes the roles of each team member.

Ahmed Hisham - Random Word Picker and Displayer

Ahmed will be responsible for the following:

Random Word Picker:

Implement the functionality for a pickRandomWord() method.

The randomization of the word has to be properly biased to work correctly.

Word Displayer:

Create a displayWord() function to visually display the current state of the word, including guessed and hidden letters.

Ensure that this display is updated correctly throughout the game.

Mahmoud Elsayed - Word Guess Logic

Mahmoud is responsible for the following:

Check Guesses:

Implement the logic in the Hangman() function to check if a guessed letter is in the word.

Mark the correct/incorrect guessed letters and prevent double guessing.

Feedback and attempts:

Give correct feedback for correct and wrong guesses.

Manage the attempts counter: update it accordingly.

Marwan Waleed - Game Flow and Winning Conditions

Marwan will handle:

Game Loop:

The game loop should be handled efficiently within the Hangman() function.

Add logic to end the game when the word is guessed or attempts run out.

Conditions for Winning:

Implement the function isWordGuessed() that checks if all the letters are correctly guessed.

Handle switching between players' turns.

Kareem Ahmed - Multiplayer Implementation and Scoring

Kareem's tasks are as follows:

Multiplayer Functionality:

The main() function will be developed with the logic of initializing the players.

Multiple players should be handled seamlessly from getting their names to turn-by-turn gameplay.

Scoring System:

Keep track and display the score of each player individually.

Ensure that the scores are updated appropriately based on the game outcomes.

Zeyad Mohamed - Testing and Debugging

Zeyad will focus on:

Testing:

Perform extensive testing of all functions, including edge cases and unusual inputs.

Validate random word selection and scoring accuracy.

Debugging:

Identify and fix any logical or runtime errors in the program.

Ensure the program handles invalid inputs gracefully.

Algorithm and external libraries

This section describes the algorithm and external libraries used in project .

Must include detailed description of them .

Algorithm

Hangman is a pretty simple game, and it also uses a quite straightforward and efficient algorithm to manage the gameplay. The key steps are as follows:

Random Word Selection:

A random word is selected from a predefined list of 26 country names.

The rand() function, seeded by the current time using srand(time(0)), ensures randomness.

Game Initialization:

The number of players and their names are taken as input.

Scores for each player are initialized to zero.

Gameplay Loop:

Each player gets to guess the letters of a randomly chosen word in turn.

The game tracks correctly guessed letters and remaining attempts.

It signals correct or incorrect guesses.

The word is dynamically shown with guessed letters and placeholders for the rest.

Game Won/Lost Conditions:

A player wins if all letters in the word are guessed within the allowed attempts.

A player loses if attempts run out before the word is guessed.

Score Management:

Scores are updated based on whether the player successfully guessed the word.

Final scores are shown at the end of the game.

External Libraries

The project uses standard C++ libraries for various functionalities:

<iostream>:

This is used for input and output operations.

It handles user input for player names and guesses.

It displays game messages, the word state, and scores.

<string>:

This provides functionality to handle strings.

Allows operations like storing and comparing guessed letters with the word.

<cstdlib>:

Contains functions such as rand() and srand() dealing with random number generation.

Used to choose a random word from the predefined list.

<ctime>:

Contains the function time(0), which provides the current time to seed the random number generator.

This guarantees that each run of a game will select a different random word.

Code explaining

Must include all code with detailed explanation.
Here is the explanation of the given code in Hangman:

---

1. Includes and Constants
```cpp
#include <iostream>
#include <string>
#include <cstdlib> // for srand and rand
#include <ctime>   // for time(0)
using namespace std;

const int attmepts = 7;
const int WordCount = 26;
const int players = 5;
```
- **Header Files**:
  - `<iostream>`: Used for input and output operations.
  - `<string>`: Allows the use of the `string` data type for words and names.
- `<cstdlib>`: Contains the `rand()` function for random number generation.
 - `<ctime>`: Enables the usage of `time()` to seed the random number generator.

- **Constants**:
  - `attmepts`: The maximum number of attempts a player has to guess the word.
  - `WordCount`: The total number of words in the word library.
  - `players`: The maximum number of players allowed.

---

2. Word Library
```cpp
const string words[WordCount] = {
"unitedstates", "china", "unitedkingdom", "france", "india", "germany", "japan",
    "russia", "brazil", "italy", "australia", "canada", "southkorea", "mexico",
    "saudi arabia", "egypt", "spain", "turkey", "argentina", "switzerland",
    "netherlands", "poland", "sweden", "southafrica", "greece", "newzealand"
};
```

- This array holds a list of words which may come up in this word game. The words here represent country names.

---

3. Selecting a Random Word
```cpp
string pickRandomWord() {
    return words[rand() % WordCount];
}
```
- **Purpose**: Retrieves a random word from the array 'words'.
- **Explanation**:
 - `rand()`: A random integer
 - `% WordCount` : This is to restrict the random number generated between 0-25 so index out of range error.
 - The picked word becomes the word of the day to be selected for playing with the user

---

4. Drawing the word
```cpp
void displayWord(const string& word, const bool guessed[]) {
for(int i = 0; i < word.length(); i++){
        if (guessed[i]) {
            cout << word[i] << " ";
        } else {
            cout << "_ ";
}
    }
    cout << endl;
}
```
- **Purpose**: Prints out the current state of the word including correctly guessed letters and underscores for the remaining letters.
- **Parameters**:
  - `word`: The word to be guessed.
  - `guessed[]`: A boolean array indicating which letters have been correctly guessed so far.
- **Example:
- In the case of the word `china`, if only the letters `c` and `h` are guessed then,
    ```
    c h _ _ _
    ```

---

5. Checking if the Word is Guessed
~~~cpp
bool isWordGuessed(const bool guessed[], int length) {
    for (int i = 0; i < length; i++) {
if (!guessed[i]) return false;
    }
    return true;
}
``
- **Purpose**: to check whether all the word letters have been guessed
- **How it works**:
   - cycles through the array `guessed[]`.
   - in case of at least one `false`, word has not yet been guessed.

---

### **6. Hangman Logic**
```cpp
int Hangman() {
string word = pickRandomWord();
    int wordLength = word.length();
    bool guessed[wordLength] = {false};
    int attemptsLeft = attmepts;
    char guess;
    bool correctGuess;

    cout << "
Guess the word! You have " << attmepts << " attempts." << endl;
~~~
- **Initialization**:
  - Picks a random word.
- This will set the word length and create an array to track the guessed letters.
- Set the number of attempts left.

Game Loop
```cpp
    while (attemptsLeft > 0) {
        displayWord(word, guessed);
        cout << "
Enter a letter: ";
        cin >> guess;

correctGuess = false;
```

- Loops until the player has no more guesses.
- Shows current word state and asks player to input a letter.

Checking Guesses
```cpp
        for (int i = 0; i < wordLength; )
if (word[i] == guess &&!guessed[i]) {
                guessed[i] = true;
                correctGuess = true;
            }
```

- Checks if the guessed letter is present in the word and marks it as already guessed.

Feedback and Attempts
```cpp
        if (correctGuess) {
            cout << "
Good guess!" << endl;
        } else {
--attemptsLeft;
            cout << "
Wrong guess! Attempts left: " << attemptsLeft << endl;
        }
```

- Informs the player of their guess.
- Decrements the remaining attempts on wrong guesses.

Winning or Losing
```cpp
if (isWordGuessed(guessed, wordLength)) {
            cout << "
Congratulations! You guessed the word: " << word << endl;
            return 1; // Player wins
        }
    }

cout << "
```

```
You lost! The word was: " << word << endl;
    return 0;
}
```
- If the word is fully guessed, the player wins.
- If attempts reach zero, the player loses.

---

7. Main Function
```cpp
int main() {
    srand(time(0));
    int numPlayers;
    int scores[players] = {0};
    string playerNames[players];

cout << "Welcome to the hangman game" << endl;
    cout << "Enter the number of players (max " << players << "): ";
    cin >> numPlayers;
```
- **Setup**:
  - Seeds the random number generator with the current time.
  - Prompts the user to enter the number of players.

Player Names
~~~cpp
for (int i = 0; i < numPlayers; i++) {
        cout << "Enter name for player " << i + 1 << ": ";
        cin >> playerNames[i];
    }
``
- Gathers the players' names.

---

#### **Player Turns**
```cpp
for (int i = 0; i < numPlayers; i++) {
        cout << "
--- " << playerNames[i] << "'s Turn ---" << endl;
        scores[i] = Hangman();
~~~

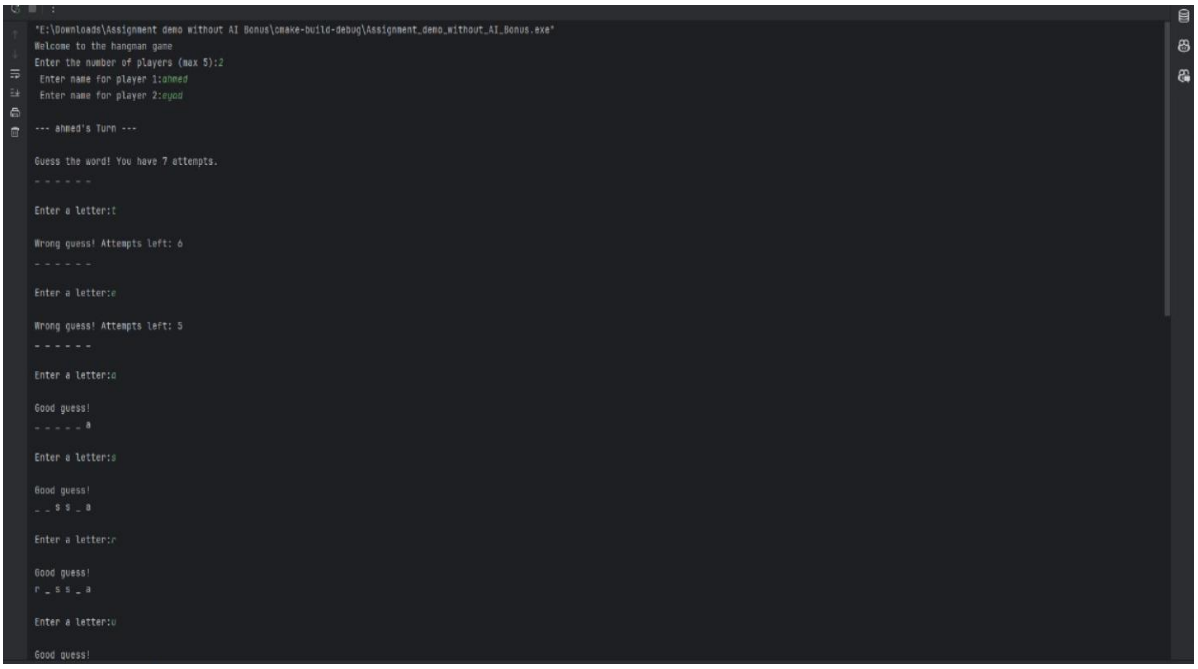
    }
``
- Each player plays a round of the game.

Final Scores
```cpp
    cout << "
--- Final Scores ---" << endl;
for (int i = 0; i < numPlayers; i++) {
        cout << playerNames[i] << ": " << scores[i] << " points" << endl;
    }

    return 0;
}
```
- Outputs the final scores of all players.

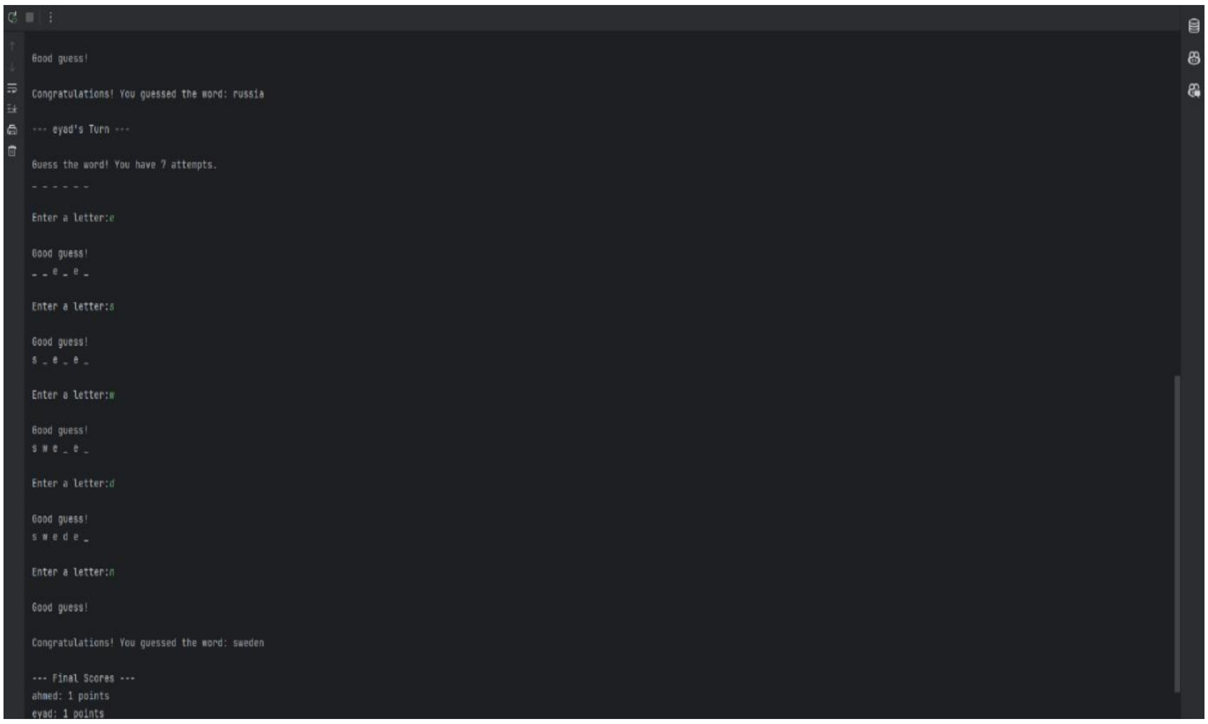
Conclusion
- Programmed a Hangman multiplayer game where players take turns guessing a word.
- It incorporates random picking words, input validation, and feedback.
- The scores keep on tracking, and there is an announcement at the end of the game.

Output and results

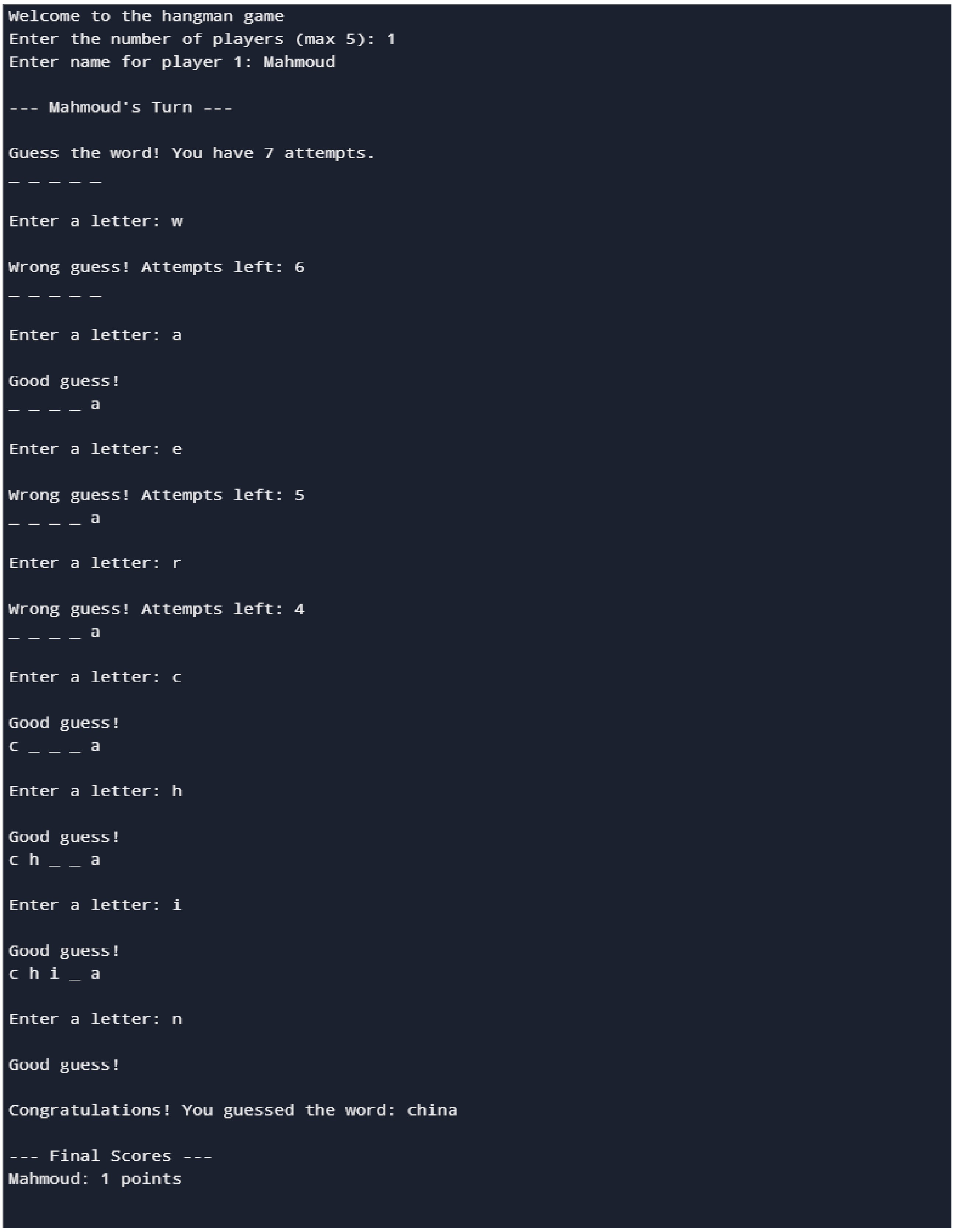
Must include the screenshots of the running program for every case with detailed explanation.



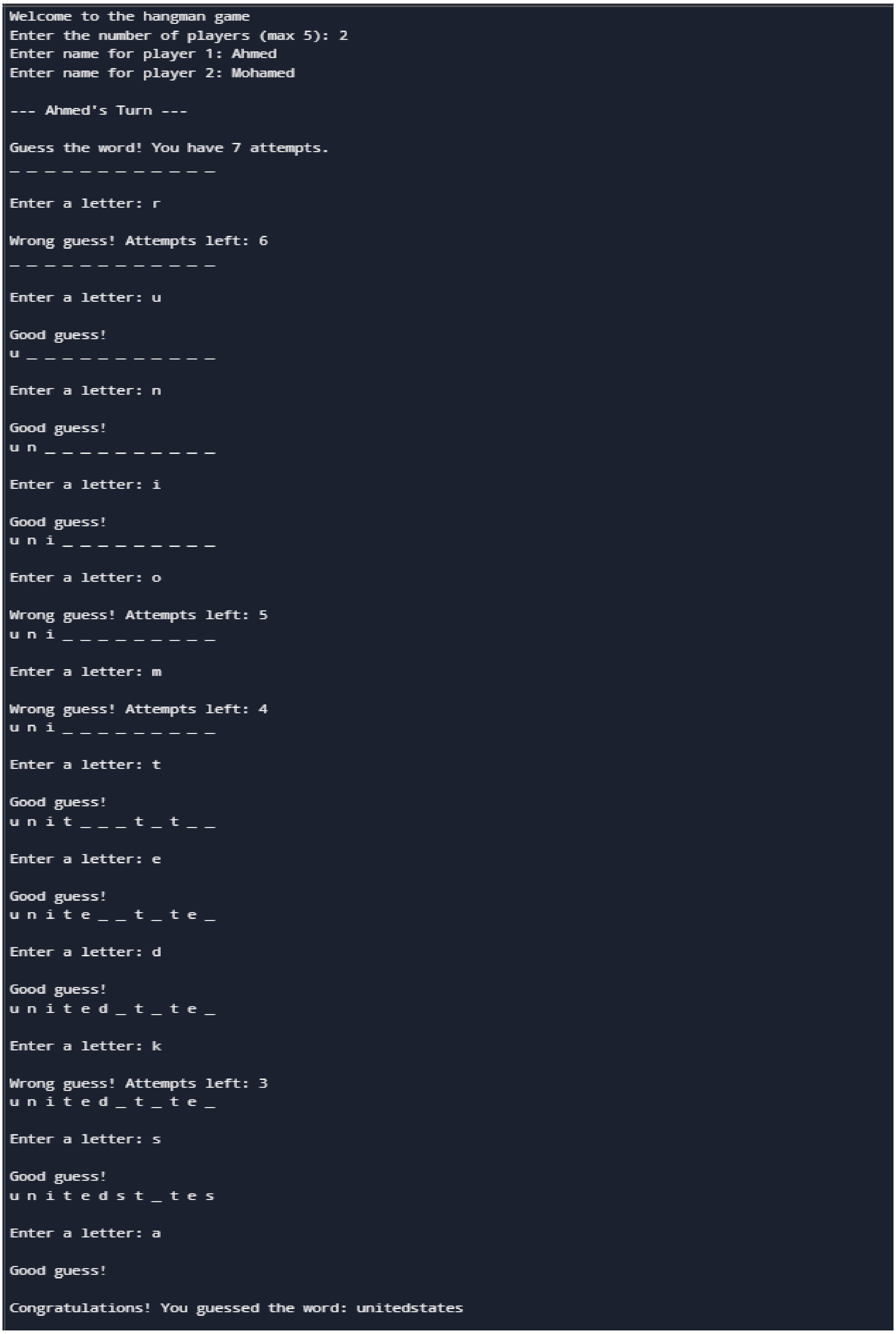
```
"E:\Downloads\Assignment demo without AI Bonus\cmake-build-debug\Assignment_demo_without_AI_Bonus.exe"
Welcome to the hangman game
Enter the number of players (max 5):2
 Enter name for player 1:ahmed
 Enter name for player 2:eyad

--- ahmed's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _

Enter a letter:t

Wrong guess! Attempts left: 6
_ _ _ _ _ _

Enter a letter:e

Wrong guess! Attempts left: 5
_ _ _ _ _ _

Enter a letter:a

Good guess!
_ _ _ _ _ a

Enter a letter:s

Good guess!
_ _ s s _ a

Enter a letter:r

Good guess!
r _ s s _ a

Enter a letter:u

Good guess!
```

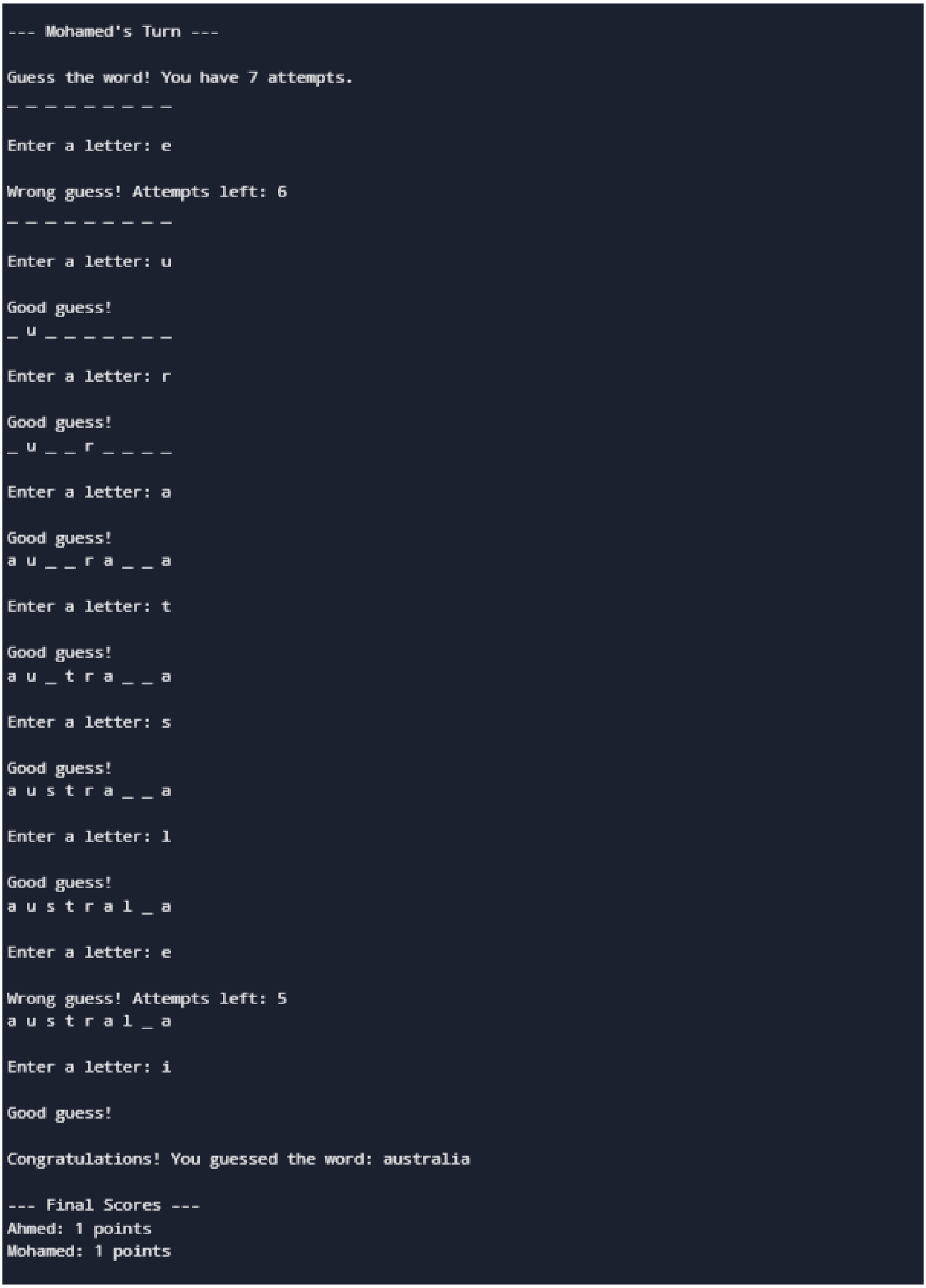


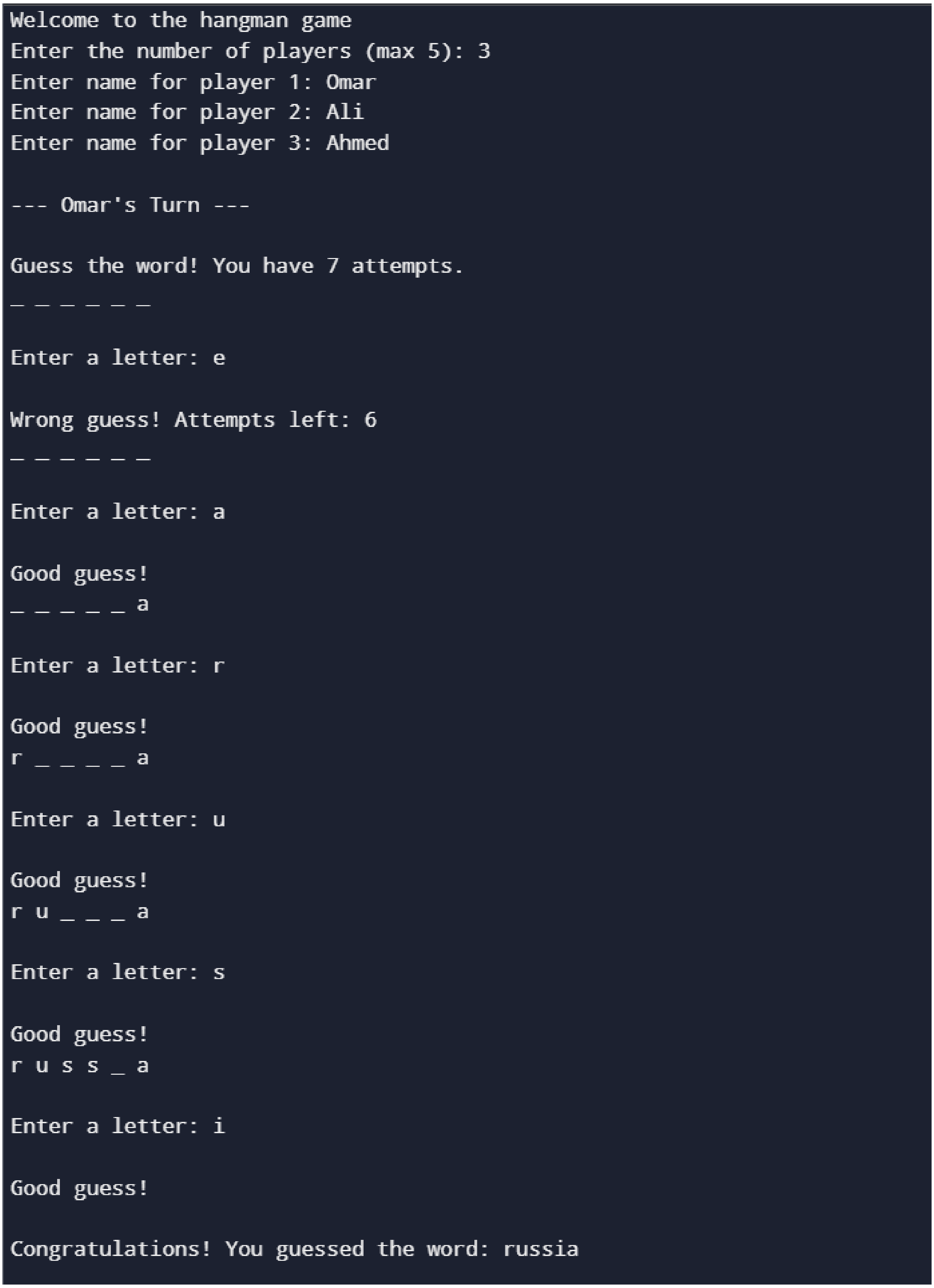
```

Good guess!

Congratulations! You guessed the word: russia

--- eyad's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _

Enter a letter:e

Good guess!
_ _ e _ e _

Enter a letter:s

Good guess!
s _ e _ e _

Enter a letter:w

Good guess!
s w e _ e _

Enter a letter:d

Good guess!
s w e d e _

Enter a letter:n

Good guess!

Congratulations! You guessed the word: sweden

--- Final Scores ---
ahmed: 1 points
eyad: 1 points
```



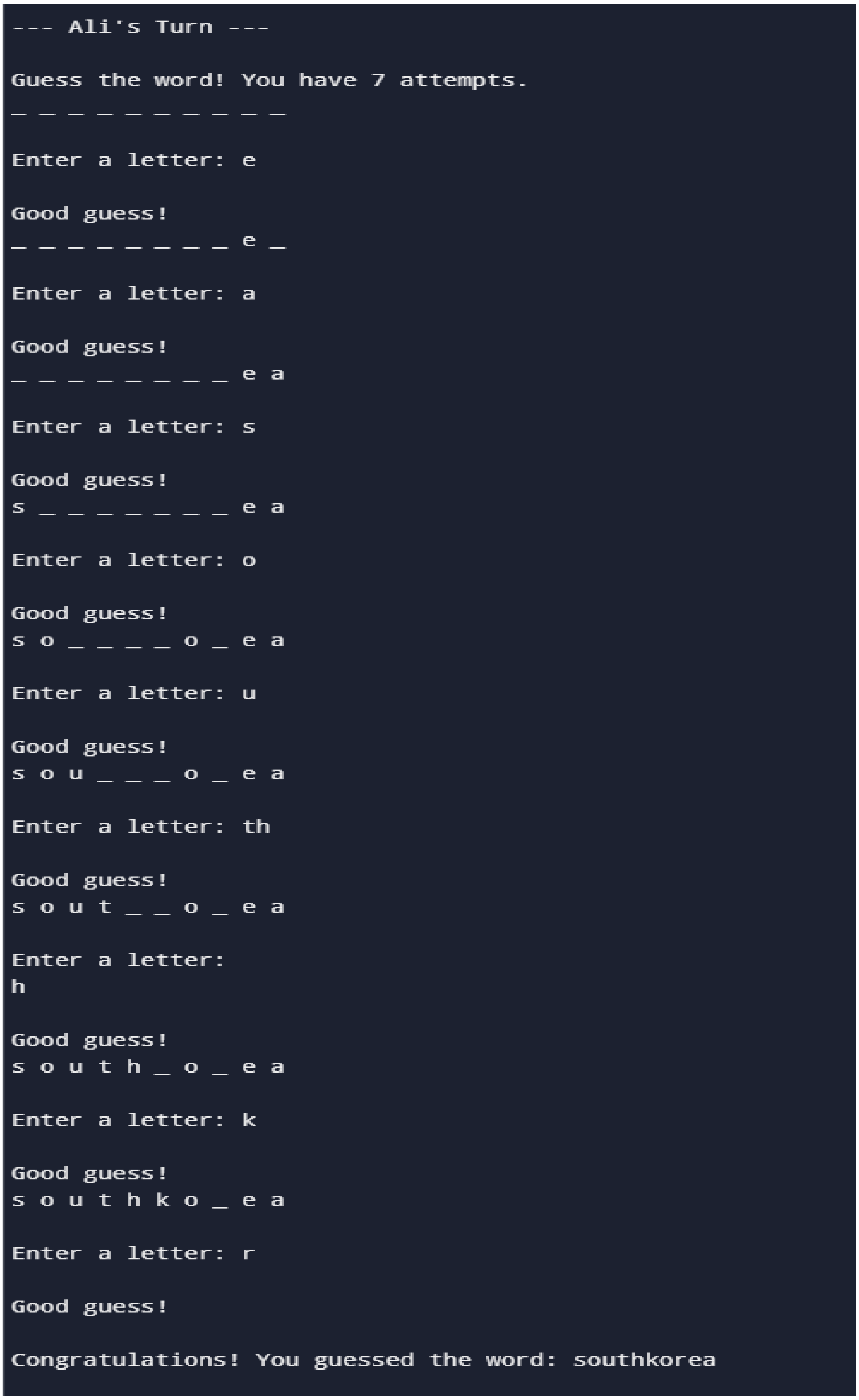
```
Welcome to the hangman game
Enter the number of players (max 5): 1
Enter name for player 1: Mahmoud

--- Mahmoud's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _

Enter a letter: w

Wrong guess! Attempts left: 6
_ _ _ _ _

Enter a letter: a

Good guess!
_ _ _ _ a

Enter a letter: e

Wrong guess! Attempts left: 5
_ _ _ _ a

Enter a letter: r

Wrong guess! Attempts left: 4
_ _ _ _ a

Enter a letter: c

Good guess!
c _ _ _ a

Enter a letter: h

Good guess!
c h _ _ a

Enter a letter: i

Good guess!
c h i _ a

Enter a letter: n

Good guess!

Congratulations! You guessed the word: china

--- Final Scores ---
Mahmoud: 1 points
```

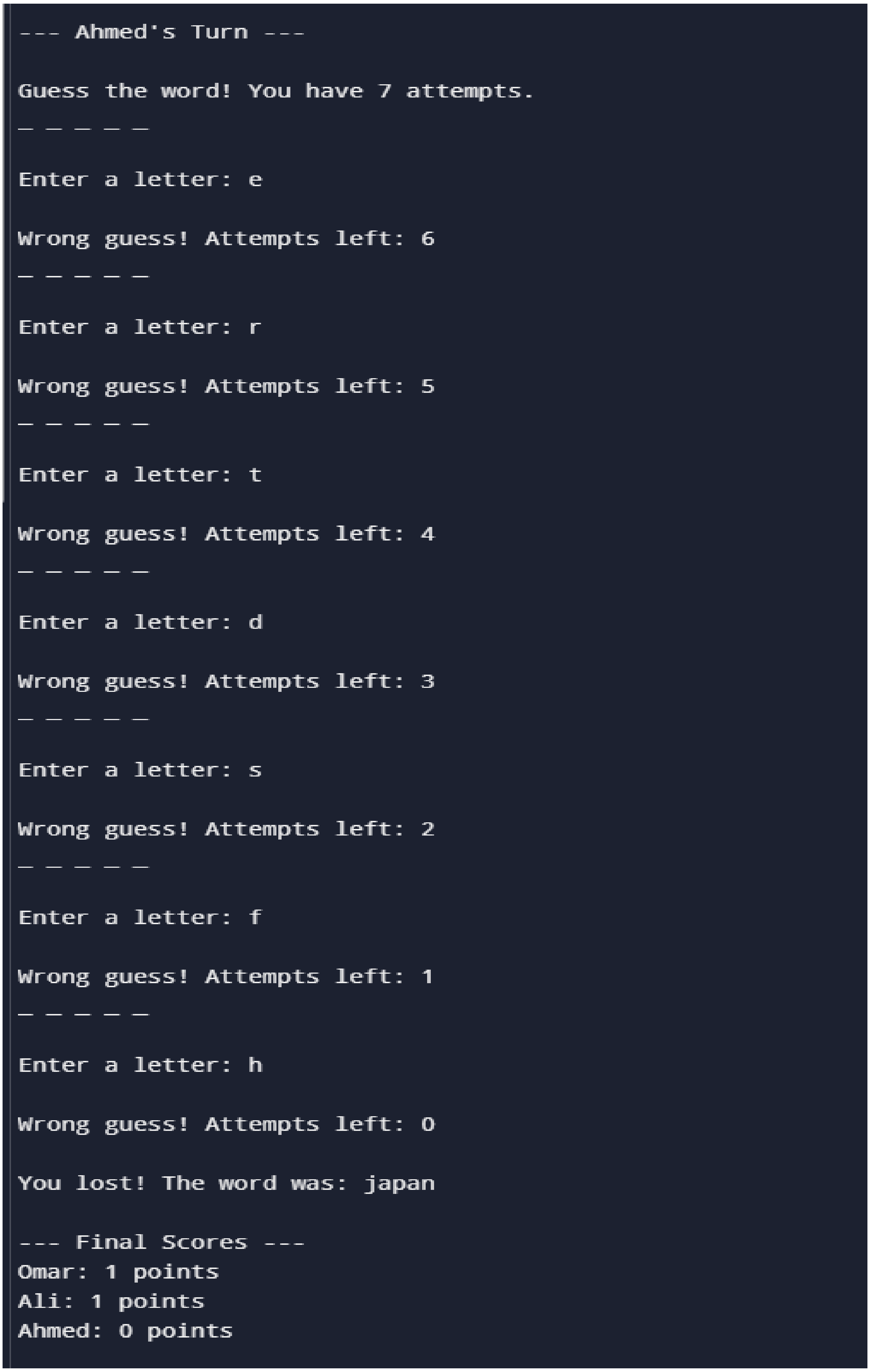

```
Welcome to the hangman game
Enter the number of players (max 5): 2
Enter name for player 1: Ahmed
Enter name for player 2: Mohamed

--- Ahmed's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _ _ _ _ _ _ _

Enter a letter: r

Wrong guess! Attempts left: 6
_ _ _ _ _ _ _ _ _ _ _ _

Enter a letter: u

Good guess!
u _ _ _ _ _ _ _ _ _ _ _

Enter a letter: n

Good guess!
u n _ _ _ _ _ _ _ _ _ _

Enter a letter: i

Good guess!
u n i _ _ _ _ _ _ _ _ _

Enter a letter: o

Wrong guess! Attempts left: 5
u n i _ _ _ _ _ _ _ _ _

Enter a letter: m

Wrong guess! Attempts left: 4
u n i _ _ _ _ _ _ _ _ _

Enter a letter: t

Good guess!
u n i t _ _ _ t _ t _ _

Enter a letter: e

Good guess!
u n i t e _ _ t _ t e _

Enter a letter: d

Good guess!
u n i t e d _ t _ t e _

Enter a letter: k

Wrong guess! Attempts left: 3
u n i t e d _ t _ t e _

Enter a letter: s

Good guess!
u n i t e d s t _ t e s

Enter a letter: a

Good guess!

Congratulations! You guessed the word: unitedstates
```


```
--- Mohamed's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _ _ _ _

Enter a letter: e

Wrong guess! Attempts left: 6
_ _ _ _ _ _ _ _ _

Enter a letter: u

Good guess!
_ u _ _ _ _ _ _ _

Enter a letter: r

Good guess!
_ u _ _ r _ _ _ _

Enter a letter: a

Good guess!
a u _ _ r a _ _ a

Enter a letter: t

Good guess!
a u _ t r a _ _ a

Enter a letter: s

Good guess!
a u s t r a _ _ a

Enter a letter: l

Good guess!
a u s t r a l _ a

Enter a letter: e

Wrong guess! Attempts left: 5
a u s t r a l _ a

Enter a letter: i

Good guess!

Congratulations! You guessed the word: australia

--- Final Scores ---
Ahmed: 1 points
Mohamed: 1 points
```

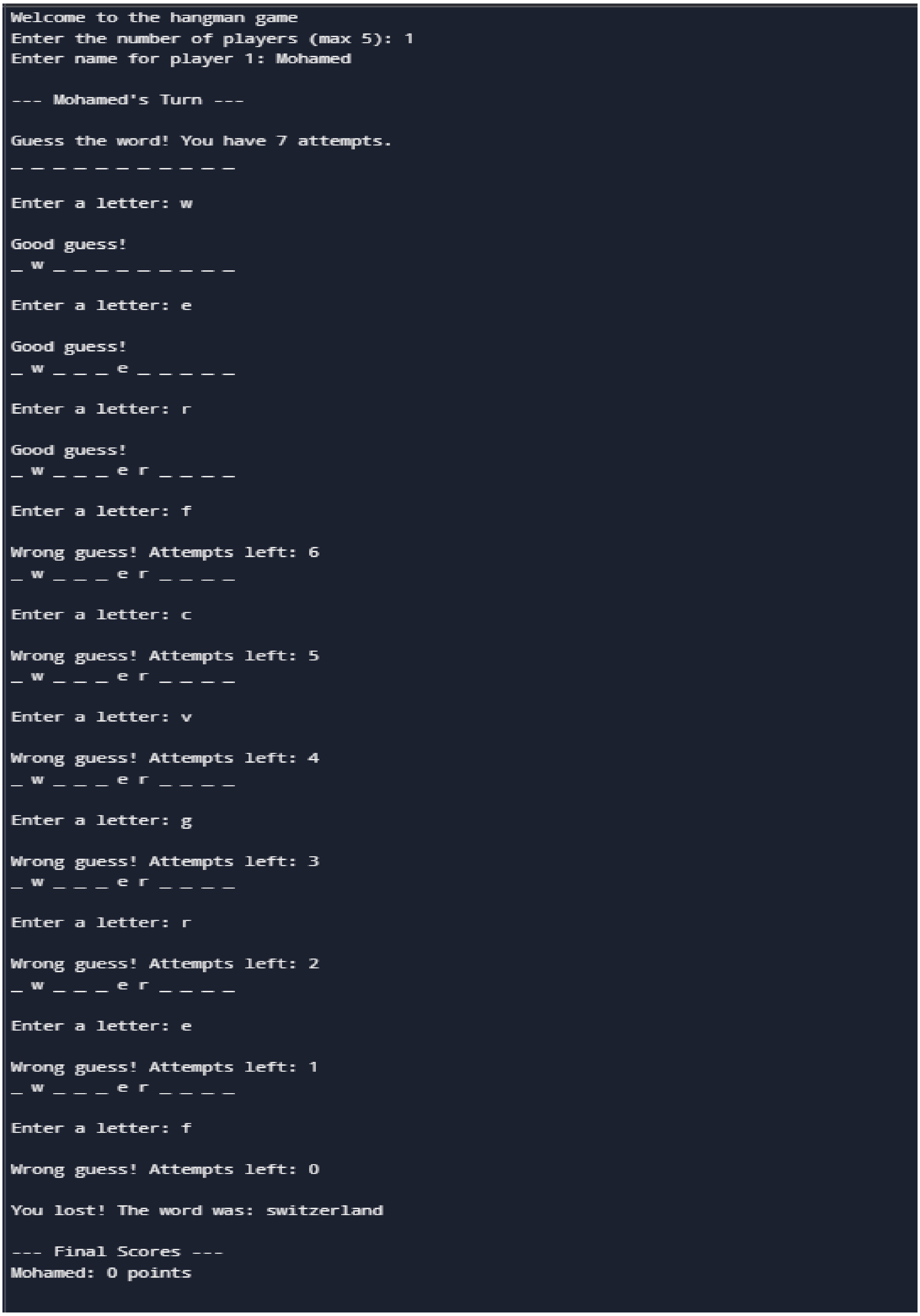


```
Welcome to the hangman game
Enter the number of players (max 5): 3
Enter name for player 1: Omar
Enter name for player 2: Ali
Enter name for player 3: Ahmed

--- Omar's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _

Enter a letter: e

Wrong guess! Attempts left: 6
_ _ _ _ _ _

Enter a letter: a

Good guess!
_ _ _ _ _ a

Enter a letter: r

Good guess!
r _ _ _ _ a

Enter a letter: u

Good guess!
r u _ _ _ a

Enter a letter: s

Good guess!
r u s s _ a

Enter a letter: i

Good guess!

Congratulations! You guessed the word: russia
```

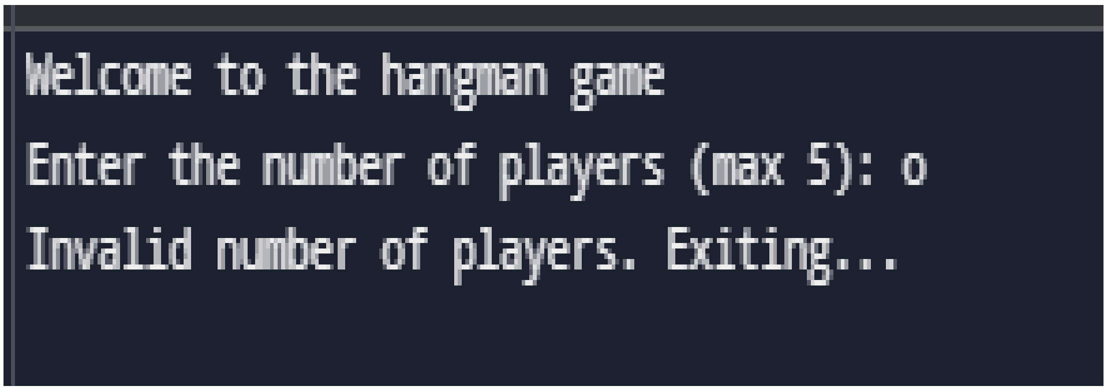

```
--- Ali's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _ _ _ _ _

Enter a letter: e

Good guess!
_ _ _ _ _ _ _ _ e _

Enter a letter: a

Good guess!
_ _ _ _ _ _ _ _ e a

Enter a letter: s

Good guess!
s _ _ _ _ _ _ _ e a

Enter a letter: o

Good guess!
s o _ _ _ _ o _ e a

Enter a letter: u

Good guess!
s o u _ _ _ o _ e a

Enter a letter: th

Good guess!
s o u t _ _ o _ e a

Enter a letter:
h

Good guess!
s o u t h _ o _ e a

Enter a letter: k

Good guess!
s o u t h k o _ e a

Enter a letter: r

Good guess!

Congratulations! You guessed the word: southkorea
```



```
--- Ahmed's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _

Enter a letter: e

Wrong guess! Attempts left: 6
_ _ _ _ _

Enter a letter: r

Wrong guess! Attempts left: 5
_ _ _ _ _

Enter a letter: t

Wrong guess! Attempts left: 4
_ _ _ _ _

Enter a letter: d

Wrong guess! Attempts left: 3
_ _ _ _ _

Enter a letter: s

Wrong guess! Attempts left: 2
_ _ _ _ _

Enter a letter: f

Wrong guess! Attempts left: 1
_ _ _ _ _

Enter a letter: h

Wrong guess! Attempts left: 0

You lost! The word was: japan

--- Final Scores ---
Omar: 1 points
Ali: 1 points
Ahmed: 0 points
```



```
Welcome to the hangman game
Enter the number of players (max 5): 1
Enter name for player 1: Mohamed

--- Mohamed's Turn ---

Guess the word! You have 7 attempts.
_ _ _ _ _ _ _ _ _ _ _

Enter a letter: w

Good guess!
_ w _ _ _ _ _ _ _ _ _

Enter a letter: e

Good guess!
_ w _ _ _ e _ _ _ _ _

Enter a letter: r

Good guess!
_ w _ _ _ e r _ _ _ _

Enter a letter: f

Wrong guess! Attempts left: 6
_ w _ _ _ e r _ _ _ _

Enter a letter: c

Wrong guess! Attempts left: 5
_ w _ _ _ e r _ _ _ _

Enter a letter: v

Wrong guess! Attempts left: 4
_ w _ _ _ e r _ _ _ _

Enter a letter: g

Wrong guess! Attempts left: 3
_ w _ _ _ e r _ _ _ _

Enter a letter: r

Wrong guess! Attempts left: 2
_ w _ _ _ e r _ _ _ _

Enter a letter: e

Wrong guess! Attempts left: 1
_ w _ _ _ e r _ _ _ _

Enter a letter: f

Wrong guess! Attempts left: 0

You lost! The word was: switzerland

--- Final Scores ---
Mohamed: 0 points
```



```
Welcome to the hangman game
Enter the number of players (max 5): o
Invalid number of players. Exiting...
```

References

1)C++ Documentation: Random Number Generation

Official C++ reference on rand() and srand() for generating random numbers.
Link: https://cplusplus.com/reference/cstdlib/rand/
C++ String Handling

2)A guide to understand std::string, its methods, and how it is used in C++.
Link: https://cplusplus.com/reference/string/string/
C++ Arrays

3)Explanation of how arrays work in C++ and how they are used for fixed-size collections like guessed[].
Link: https://cplusplus.com/doc/tutorial/arrays/
C++ Boolean Data Type

4)Documentation on the bool type, used for the guessed[] array.
Link: https://cplusplus.com/doc/tutorial/variables/
C++ Input/Output Basics

5)Understanding the input and output streams, cin and cout, utilized throughout the code. Link:
https://cplusplus.com/doc/tutorial/basic_io/