**Data Structures & Algorithm Lab**

**(2+1 Credit Hours)**

CSL-221

**Project Proposal**

**Project Title: Chess Game**

**Submitted by**

Sara Tariq

**DEPARTMENT OF COMPUTER SCIENCE** BAHRIA UNIVERSITY, KARACHI, PAKISTAN

1. **Introduction:**

In the world of strategy and intellect, few games can match the timeless elegance and complexity of chess. Chess is a two-player board game that challenges players to outwit and outmaneuver their opponents through careful planning and strategic thinking. With a rich history spanning over a thousand years, chess has captivated minds and inspired countless players around the globe.

In this project, we will explore the implementation of a chess game using the C++ programming language, incorporating fundamental Data Structures and Algorithms (DSA) concepts. By building our own chess engine, we will delve into the intricacies of the game, from representing the board state to implementing the rules and strategies that govern each move. Throughout this project, we will leverage key DSA principles to design efficient algorithms for move generation, board evaluation, and game tree search. We will employ data structures such as arrays, linked lists, and trees to manage the chessboard, store the game history, and evaluate potential moves to optimize our chess engine's performance and enhance its decision-making capabilities.

By developing this chess game using C++ and DSA, we aim to provide an immersive and interactive experience for both

1. **Objective:**

* **Implement the basic structure and functionality of a chess game:** The primary objective is to create a functioning chess game that allows two players to play against each other. This includes setting up the chessboard, representing the pieces, and implementing the rules of movement and capture.
* **Develop data structures for board representation:** Design and implement data structures to represent the chessboard and its pieces. This may involve using arrays, linked lists, or other suitable data structures to efficiently store and manipulate the state of the game.
* **Implement move generation and validation:** Develop algorithms to generate and validate legal moves for each piece on the board. This includes considering piece-specific movement rules, checking for obstructions or captures, and enforcing the game's overall rules.
* **Implement game logic and player turns:** Establish the game flow by implementing logic to alternate player turns, validate moves, and update the board accordingly. Ensure that the game enforces the rules of chess, such as checking for checkmate, stalemate, and promoting pawns to higher-ranked pieces.
* **Test and debug the chess engine**: Thoroughly test the implemented functionalities, identifying and fixing any bugs or logical errors. Validate the correctness of the game's behavior and ensure that it conforms to the rules of chess.

By accomplishing these objectives, we aim to create a robust and feature-rich chess game in C++ that demonstrates the application of DSA concepts. The end result will be an engaging chess-playing experience that showcases the beauty and complexity of the game while providing opportunities for further exploration and improvement.

1. **Problem Description:**

Building a chess game using C++ and DSA involves addressing several challenges. These include efficiently representing the chessboard and its pieces, implementing move generation and validation for various piece types, managing game logic and player turns, developing intelligent decision-making, implementing additional features like castling and passant capture, optimizing the game's performance, and thoroughly testing and debugging the chess engine. Successfully overcoming these problems requires a deep understanding of chess rules, proficiency in data structures and algorithms, and careful consideration of edge cases and optimizations.

1. **Features:**

A chess game in C++ using DSA can have a variety of features, depending on the complexity of the game. Some common features include:

* **Two-player mode:** The game can be played by two human players.

Computer opponent: The game can be played against a computer opponent.

* **Graphical user interface:** The game can have a graphical user interface (GUI) that makes it easy to play.
* **Tutorial:** The game can have a tutorial that teaches players how to play chess.

The features of a chess game in C++ using DSA can be further enhanced by using more advanced data structures and algorithms. For example, a game that uses a hash table to store the board state can search for possible moves more quickly than a game that uses a linked list.

1. **Methodology**

. The methodology for building a chess game using C++ and DSA involves several steps and approaches. Here is a high-level overview of the methodology:

* Begin by familiarizing yourself with the rules and mechanics of chess. Gain a comprehensive understanding of how the pieces move, special moves like castling and en passant, promotion, and rules related to checkmate and stalemate.
* Outline the overall structure of your chess game, including the classes, data structures, and algorithms you will use. Determine how the game logic will be implemented.
* Choose an appropriate data structure to represent the chessboard. This could be a 2D array, a linked list, or a combination of data structures. Implement the necessary methods to initialize the board, update piece positions, and retrieve information about the board state.
* Develop algorithms to generate and validate legal moves for each piece. Consider the movement rules for each piece type and handle special moves and restrictions. Validate moves based on the current board state, check for obstructions, and enforce rules like castling and pawn promotion.
* Design and implement the game logic to manage player turns, handle move validation, and enforce rules.
* Thoroughly test the implemented functionalities and handle edge cases. Conduct extensive testing to ensure the correctness and reliability of the chess engine. Debug any issues or logical errors that arise during testing.

1. **Project Scope:**

The project scope of a chess game in C++ in dsa can be defined as the following:

* The game should be able to be played by two players. Each player should be able to control their own set of pieces.
* The game should follow the rules of chess. This includes the movement of pieces, the capture of pieces, and the checkmate of the king..

The project scope can be further refined by adding additional features, such as:

* A graphical user interface. This would make the game more user-friendly.
* A tutorial mode. This would teach new players how to play chess.

1. **Tools/Technology:**

we are using tools for programming:

* Visual studio code
* Dev C++
* Visual Studio

And there are many libraries available that can be used to simplify the process of programming a chess game.

1. **Milestones:**

In this milestone, our objective is to create a functional and user-friendly chess game that enables two players to engage in a competitive and enjoyable gaming experience. The key objectives include conducting research on chess rules and mechanics, designing an intuitive user interface with a visually appealing chessboard layout, implementing the game logic to enforce the rules and generate valid moves, enabling player interactions for piece selection and movement, incorporating win conditions and game over scenarios, conducting thorough testing and debugging, and finalizing the game with comprehensive documentation. By achieving these milestones, we will deliver a successful and engaging chess game for our users.