VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

ĐẠI HỌC TÔN ĐỨC THẮNG
TON DUC THANG UNIVERSITY

THANT THIRI MAUNG - 522K0050

YAMIN THIRI WAI - 522K0046

FINAL REPORT
FUNCTIONAL PROGRAMMING

...CODE: 505011...

HO CHI MINH CITY, 2023

VIETNAM GENERAL CONFEDERATION OF LABOR

TON DUC THANG UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

THANT THIRI MAUNG - 522K0050

YAMIN THIRI WAI - 522K0046

FINAL REPORT
FUNCTIONAL PROGRAMMING

. . . CODE: 505011. . .

Advised by

Dr. Do Nhu Tai

HO CHI MINH CITY, 2023

ACKNOWLEDGEMENT

We extend our heartfelt thanks to Dr.Do Nhu Tai for his outstanding guidance during the Functional Programming Course. His passion, expertise, and support have greatly enriched our learning experience. We are grateful for the valuable insights and inspiration that will undoubtedly shape our future pursuits in Functional Programming.

Ho Chi Minh City, 24th December 2023.
Authors
(Signature and full name)

Thant
Thant Thiri Maung

Yamin
Yamin Thiri Wai

DECLARATION OF AUTHORSHIP

We hereby declare that this is our own project and is guided by Dr. Do Nhu Tai; The content research and results contained herein are central and have not been published in any form before. In addition, the project also uses some comments. If something wrong happens, We'll take full responsibility for the content of my project. Ton Duc Thang University is not related to the infringing rights, the copyrights that We give during the implementation process (if any).

Ho Chi Minh City, 24th December 2023.
Authors
(Signature and full name)

Thant
Thant Thiri Maung

Yamin
Yaming Thiri Wai

ABSTRACT

We extend our heartfelt thanks to Dr.Do Nhu Tai for his outstanding guidance during the Functional Programming Course. His passion, expertise, and support have greatly enriched our learning experience. We are grateful for the valuable insights and inspiration that will undoubtedly shape our future pursuits in Functional Programming.

Contents

List of Figures

0.1 Introduction

The Memory Game is an interactive puzzle that challenges players to test and improve their memory skills. In this implementation, players are presented with a grid of symbols that are initially hidden. The objective is to uncover pairs of matching symbols by revealing two cells at a time. The game continues until all pairs are successfully matched.

0.2 Purpose of the Report

This report aims to provide an overview and documentation of the Memory Game program. Throughout the report, various aspects of the code, including its organization, features, and user interaction, will be discussed in detail.

0.3 Section 1: Analysis of Requirements and Constraints

0.3.1 Game Objective :

The primary objective of the Memory Game is to challenge players' memory. Players are tasked with uncovering pairs of matching symbols hidden within a grid of cells. The game requires users to remember the locations of symbols as they make successive moves, aiming to reveal and match all pairs within the specified number of moves.

0.3.2 Constraints :

- Board Size Limitation: User-defined height and width determine the board size, limiting available cells for symbols.
- Symbol Range: Symbol pairs (k) constrain the symbol range, ensuring enough unique symbols for pairs without exceeding total board cells.
- User Interaction: Interactive gameplay relies on user input for cell selection, enforcing user-friendly moves.

0.3.3 Scope of Work :

- Board Initialization: Generates a random board based on user dimensions and symbol pairs.
- Game Flow: Guides players through moves, validating and updating the board with match feedback.
- User Interaction: Prompts users for moves (row and column indices), integrating a time delay (t) for an enhanced gaming experience.
- Game Completion: Congratulates players upon successfully matching all symbol pairs.

0.4 Functions for Display and Interaction

0.4.1 Board Display :

- Header Information: Displays column indices for move reference.
- Grid Lines: Horizontal and vertical lines organize the board.

- Cell Representation: Each cell exhibits a symbol and reveal status ('*' if unrevealed).
- Row and Column Numbering: Numeric indices aid in cell selection.
- Delimiter Lines: Delimiter lines separate rows, enhancing the readability of the board.

0.4.2 User Iteraction :

- Move Input: Players input moves with row and column indices.
- Cell Validation: The selected cell's validity is checked ensures valid cell selection.
- Outcome Display: Informs players of move outcomes.
- Delay Feature: 'delaySeconds' introduces interactive pauses.
- Game Flow Control: 'playGame' manages game progression until completion.

0.5 Flowcharts and Pseudocode

0.5.1 Flowcharts :

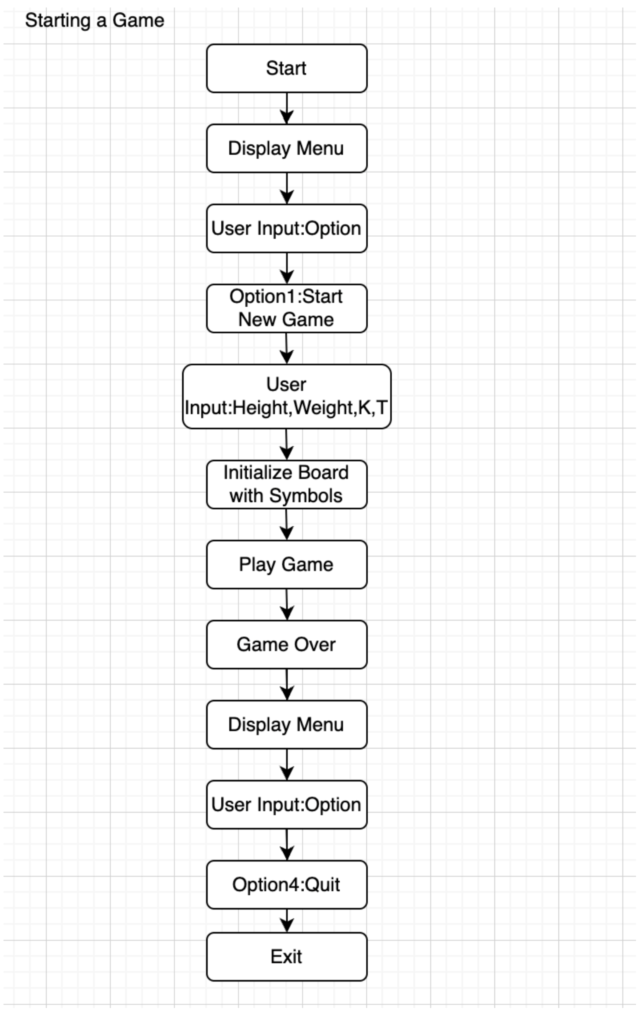


Figure 1: Starting the Game

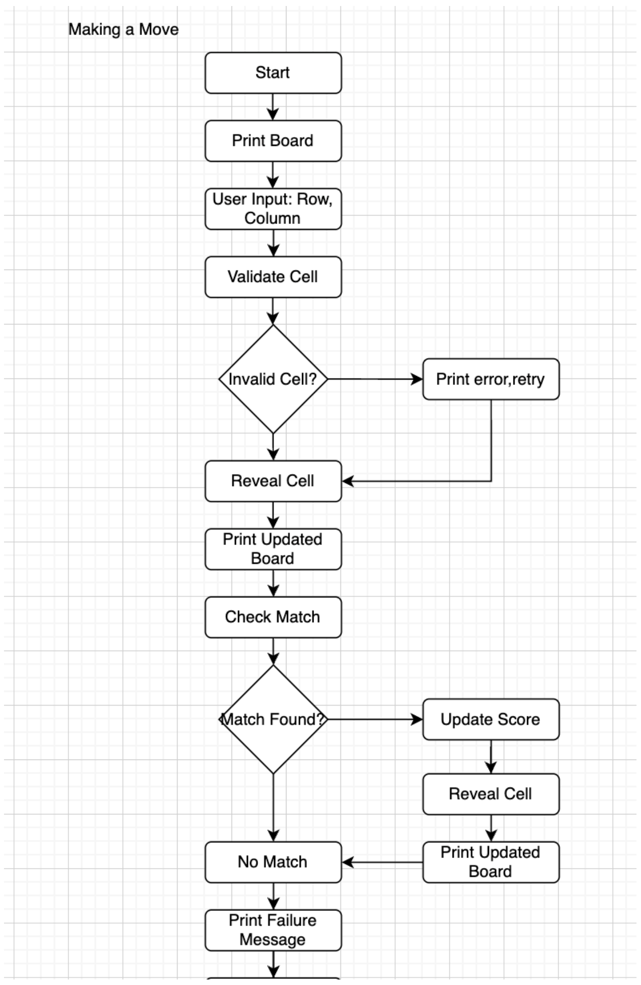


Figure 2: Making a Move 1

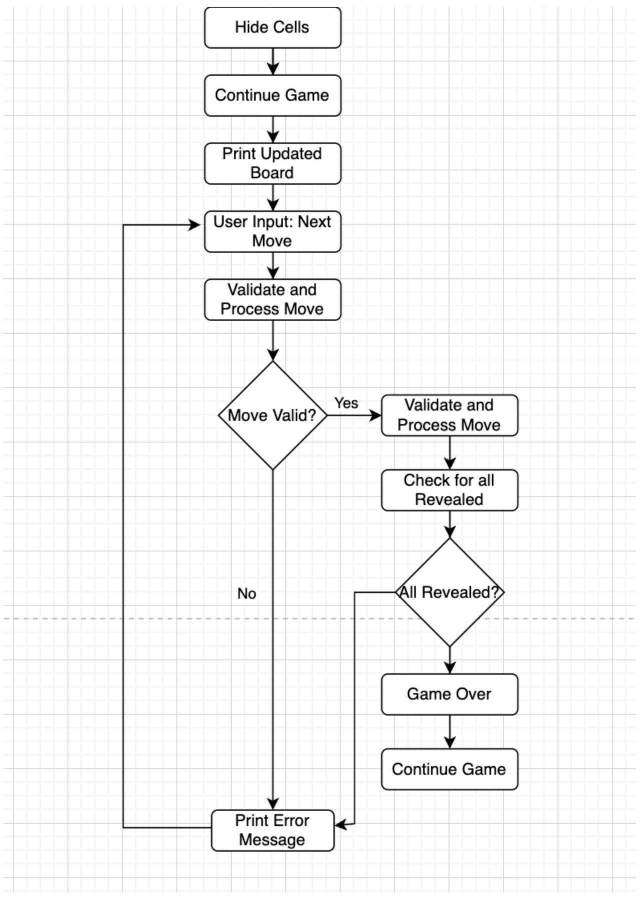
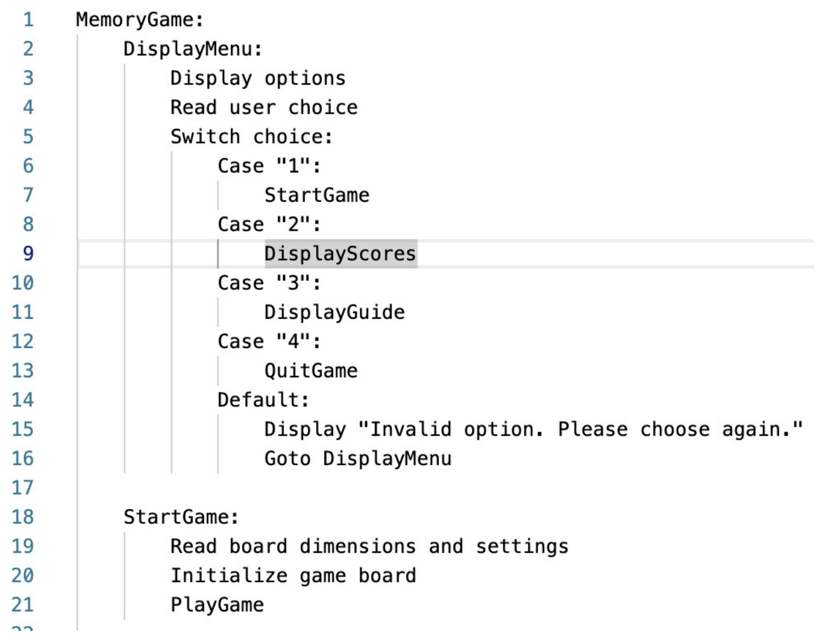


Figure 3: Making a Move 2

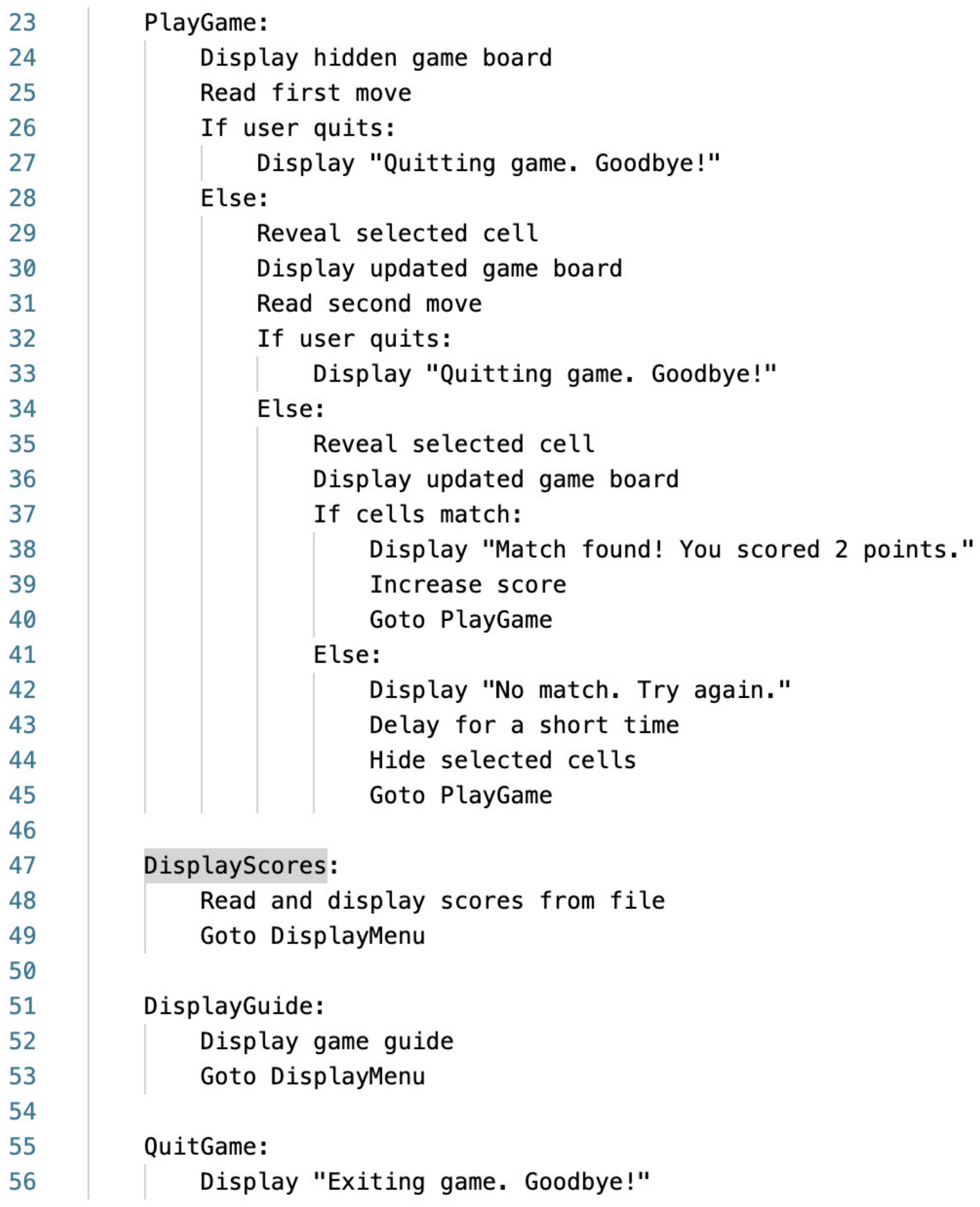
0.5.2 Pseudocode :



```
MemoryGame:
    DisplayMenu:
        Display options
        Read user choice
        Switch choice:
            Case "1":
                StartGame
            Case "2":
                DisplayScores
            Case "3":
                DisplayGuide
            Case "4":
                QuitGame
            Default:
                Display "Invalid option. Please choose again."
                Goto DisplayMenu

    StartGame:
        Read board dimensions and settings
        Initialize game board
        PlayGame
```

Figure 4: Pseudocode Part 1



```
PlayGame:
    Display hidden game board
    Read first move
    If user quits:
        Display "Quitting game. Goodbye!"
    Else:
        Reveal selected cell
        Display updated game board
        Read second move
        If user quits:
            Display "Quitting game. Goodbye!"
        Else:
            Reveal selected cell
            Display updated game board
            If cells match:
                Display "Match found! You scored 2 points."
                Increase score
                Goto PlayGame
            Else:
                Display "No match. Try again."
                Delay for a short time
                Hide selected cells
                Goto PlayGame

DisplayScores:
    Read and display scores from file
    Goto DisplayMenu

DisplayGuide:
    Display game guide
    Goto DisplayMenu

QuitGame:
    Display "Exiting game. Goodbye!"
```

Figure 5: Pseudocode Part 2

0.6 Message Printing and Board Status Display

0.6.1 Message Printing :

- Game Guides: Initial instructions guide moves and provide essential information.
- Winning Messages: After completing the game, a congratulatory message, including move count, is displayed.
- Error Messages: Invalid moves trigger error messages, enhancing user awareness.

0.6.2 Board Status Display :

- Updated Board: After each move, the board reflects current revealed and hidden cells, tracking game progress.
- Symbol Display: Symbols aid decisions; revealed symbols show, unrevealed cells display placeholders.
- Score Information: Post-match, the display may show the player's score, offering instant move impact feedback.

0.7 Data Structure for Game Information

0.7.1 The Memory Game utilizes two key data structures :

- **'Cell' Data Structure**: - Definition: 'type Cell = (Char, Bool)' representing a symbol and its revelation status. - Suitability: Suitable for encapsulating symbol and status within a single unit, simplifying representation and manipulation of each grid cell.
- **'Board' Data Structure :** - Definition: 'type Board = [[Cell]]' representing the 2D array of cells. - Suitability: Ideal for managing the game state, facilitating spatial relationships, and providing flexibility for dynamic board configurations.

0.8 User Menu and Functionalities

0.8.1 Menu Options :

- **Start New Game :**
  - Functionality: Initiates the process of starting a new game.
  - User Interaction: Prompts the user to input parameters like board dimensions, symbol pairs, and time delay.
- **Scores :**
  - Functionality: The "Scores" option allows users to view their previous game scores. The functionality displays the scores achieved in previous games.
  - User Interaction: Upon choosing this option, the program reads and displays the scores from the "scores.txt" file. Users can now see their performance history in the Memory Game.
- **Guide :**
  - Functionality: Provides a guide on how to play the Memory Game.

  - User Interaction: Displays instructions and tips for navigating through the game.

- **Quit :**
  - Functionality: Exits the game.
  - User Interaction: Prints a farewell message and terminates the program.

0.8.2 Game Information and Scores :

- **Viewing Scores :**
  - Functionality: Currently a placeholder; the code lacks the implementation to display scores.
  - User Interaction: Selecting the "Scores" option informs the user that the scores display is not yet available.
- **Scores Implementation :**
  - Future Improvement: Scores could be implemented by storing historical game scores in a file.
  - User Interaction: The "Scores" option would then display past game achievements, providing a sense of progress.

0.9 Group Work Assignment

1. **Leader(Thant Thiri Maung) :**
   - Took responsibility of the code implementation, focusing on critical components.
   - Assumed a leadership role in coordinating the overall project.
   - Conducted thorough reviews of the documentation to ensure accuracy and completeness.
   - Collaborated with team member to address identified issues.
2. **Team Member(Yamin Thiri Wai) :**
   - Focused on debugging tasks, resolving issues, and improving code quality.
   - Contributed to the documentation by providing insights into challenges.
   - Worked closely with the Leader to ensure the code aligns with project requirements.
   - Shared responsibilities for certain documentation sections.

0.10 Achieved Results and Program Evaluation

0.10.1 Results :

- **Board Initialization :**
  - Successful generation of a dynamic game board based on user input.
  - Random distribution of symbols for an engaging gaming experience.
- **Game Flow :**

  - Seamless progression through player-initiated moves.
  - Real-time feedback on match success or failure.
- **User Interaction :**
  - Prompted input through row and column indices.
  - Utilization of the 'read' function for enhanced user interaction.
- **Game Completion :**
  - Successful conclusion upon matching all symbol pairs.
  - Congratulatory message and return to the main menu

0.10.2 Evaluation :

- **User-Friendliness:**
  - Positive user interaction with room for clearer error messages.
- **Game Dynamics :**
  - Effective symbol distribution and dynamic board initialization.
- **Documentation and Debugging :**
  - Collaborative roles enhance code quality and documentation.
- **Scalability :**
  - Handles variations in board size and symbol pairs effectively.
- **Enhancement Opportunities :**
  - Potential for refining error messages and introducing a scoring system.
  - Room for future enhancements to enrich the gaming experience.

0.11 Conclusion

The Memory Game project successfully implements core functionalities, delivering an engaging user experience. Roles division, with the leader overseeing code and documentation, and the other focusing on debugging and writing, enhances collaboration. The game effectively challenges memory skills and provides a user-friendly interface. Positive aspects include dynamic gameplay, while areas for refinement include error messaging. Overall, the project lays a strong foundation for future enhancements, making it a successful and promising endeavor.

The completed Memory Game implementation in Haskell can be found in our GitHub repository: [https://github.com/thantthirimaung/Memory-Game].