

System Requirements Specification

November 24, 2014

Mancala

Anjan Kumar Patel

Harshank Shrotriya

Pulkit Ghoderao

Submitted in partial fulfilment of the requirements of CS 101: Computer Programming

TABLE OF CONTENTS

Topics	Page
Introduction	
(a)Purpose	3
(b)References	3
Overall Description	
(a)Project Perspectives	3
(b)Features	3
(c)User Class	3
(d)Operating Environment	3
(e)User Documentation	4
Specific Requirements	
(a)External Interface Requirements	4
(b)Functional Requirements and Algorithm	4
(c)Non-Functional Requirements	6

1.0 Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the requirements needed for building a c++ version of the game Mancala. It will explain the features of the project and outline what the program will do under what circumstances.

1.2 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.
http://perceval.gannon.edu/xu001/teaching/shared/sommerville/templates/srs_template.doc.

2.0 Overall Description

2.1 Project Perspectives

This is a new self-contained project and is not based upon any other project.

2.2 Features

The game interface should allow the user to choose between different menu items such as:

1. Play
 - 1.1 Single Player
 - 1.2 Double Player
2. Instructions
3. Exit

Also, a graphics interface which displays the Mancala board and updates it after user provided stimulus must be present.

2.3 User Class

Mancala will have one user class, the player/s. Members of this user class may be from ages seven to adult and will have basic computer skills.

2.4 Operating Environment

The game has been designed on and to work best with Code :: Blocks Integrated Development Environment.

The game will successfully execute on any operating system which supports the Code::Blocks IDE integrated with simplecpp .

2.5 User Documentation

User help files will be available from the game itself. The help files will explain the rules of the game as well as the layout of the user controls.

3 Specific Requirements

3.1 External Interface Requirements

The external requirements for the project include keyboard, mouse, Code::Blocks environment with integrated simplecpp. (refer: user draft manual for more details on the same).

The Canvas API.

3.2 Functional Requirements and Algorithm

(Note: In this section an integrated functional requirements specification and code algorithm are presented which is in violation of IEEE standards.)

General Algorithm

The number of seeds contained in each house and the mancalas should be stored in an array. Where each array location refers to particular houses and mancalas.

Generate a function which has the following properties

3.2.1 //PRECONDITION : Given the board and the playing array

Synchronise between the board and the value of seeds in each array and update the board accordingly.

//POSTCONDITION : After a change in the contents of the playing array the board must be updated accordingly.

3.2.2 //PRECONDITION : Given a click on the board

Check whether the click is valid or not.

//POSTCONDITION : After registering the click, function must return true or false.

3.2.3 //PRECONDITION : Given that 3.2.1 has successfully executed

Free the dynamically allocated memory in function 3.2.1.

//POSTCONDITION : The array must be deleted from heap successfully.

3.2.4//PRECONDITION : After selecting play single player option.

Play the single player successfully by invoking all other functions and in function code.

//POSTCONDITION : Single player executes successfully.

3.2.5//PRECONDITION : After selecting play double player option.

Play the double player successfully by invoking all other functions and in function code.

//POSTCONDITION : Double player executes successfully.

3.2.6//PRECONDITION : Given the playing array at any instant.

Apply the strategy for getting best move.

//POSTCONDITION : Return the number of the array element that needs to be played in order to achieve maximum profit.

3.2.7//PRECONDITION : Given a temporary playing array.

Check whether free turn or capture is possible for the player. To be used in 3.2.4 as first deepness.

//POSTCONDITION : Return true or false according to its definition.

3.2.8//PRECONDITION : Given the playing array at any given time.

Return the value of the house where free turn for the player is possible. To be used in 3.2.4.

//POSTCONDITION : Return an integer according to its definition.

3.2.9//PRECONDITION : Given the playing array at any given time.

Return the value of the house where capture for the player is possible. To be used in 3.2.4

//POSTCONDITION : Return an integer according to its definition.

3.2.10//PRECONDITION : Given the board and that the game is over.

Ask for the name of the player/s. Compare if it is a relatively (by ratio) high score. Store the names and score in a file.

//POSTCONDITION : Successfully save the high score.

3.2.11//PRECONDITION : Given that high score option is selected.

Display the high score.

//POSTCONDITION : Perform according to its definition.

3.2.12 Main function

Help to choose between various playing options. Call to all other functions according to menu item selected by the user.

Execute the entire game successfully.

3.3 Non-Functional Requirements

3.3.1 Performance Requirements

Game Board

The game board interface must respond to user action within minimal (approx. 2 seconds) time.

It must also allow the facility to ignore multiple clicks.

The minimum time required for user to move next must not exceed 5 seconds.

After a two player game is started, the interface must change itself accordingly, indicating the updated player names.

3.3.2 Quality Attributes

Portability

The game must run on any IDE similar to Code::Blocks irrespective of the operating system provided that the IDE adheres to the machine (on which the game is to be played) specifications perfectly.

Usability

The game presumes a basic level of IQ .The instructions should help the first time user to understand gameplay to a good degree after three to five games.

END OF SRS