Tic-Tac-Toe Report

June 29

Samuel Pitchforth

Contents

Defining and understanding the problem

Software description

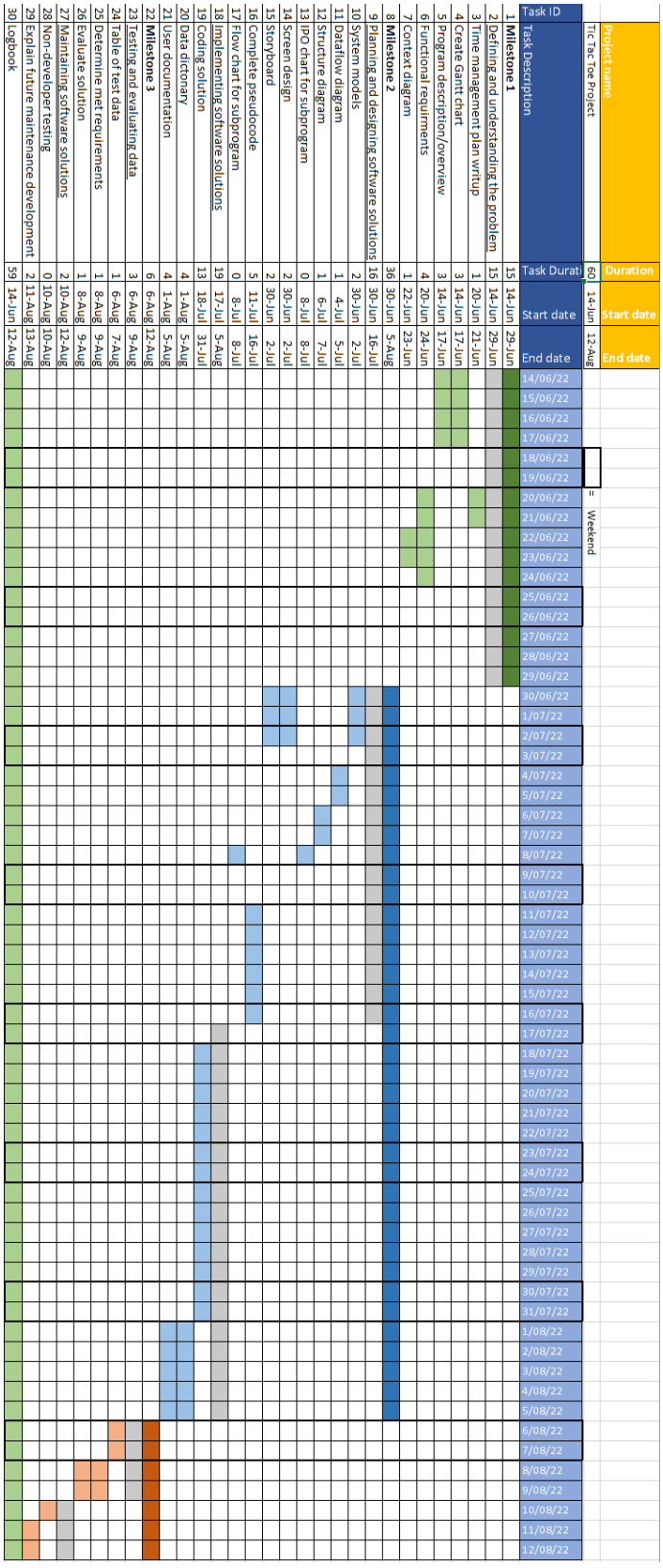
The software required is a text-based game of Tic-Tac-Toe against a computer-controlled opponent. The game takes place on a 4*4 size board and the players take turns placing their counters (“X” or “O”) on an empty square. The user always starts first and the first player to create an unbroken line 4 counter long wins. At this point the game ends and the winner is conveyed. Alternatively, if there is no space left on the board and neither player has won, it is a draw. This game must be password protected and include a main menu with options to play the game, read instructions, and exit the program. The only way to exit is from the main menu, every other option returns to the main menu when completed.

Functional Requirements

- Start with a security screen
- Allow up to 4 password attempts
- If the correct password (1234) is entered open the main menu
- If the correct password is not entered, communicate what happened to the user, then exit the program
- The main menu has a welcome message/title
- The main menu has 3 choices: Player Vs Computer, Instructions, Exit
- By entering 1, 2 or 3, the user can choose the corresponding option
- If any other input is given, an error message should be printed and the entire main menu should be reprinted including the input prompt
- After the Player Vs Computer or the Instructions mode have run and then ended, the main menu should be reprinted
- Selecting “Player Vs Computer” begins the game between the user and a computer-controlled opponent
- The player always goes first and is assigned the “X” counter

- As the game begins, the user will be prompted to enter their name which is stored for later use
- Whenever the turn is passed to the user, they will be prompted to make a move with their name
- The game starts with an empty board of the dimensions 4*4
- The board state is stored as a 1D array
- Whenever it is the user's turn, they are prompted to enter a move. A valid input consists of a number in the range 1-16 inclusive
- If an invalid input is entered, an appropriate error message is displayed and the user is prompted for a valid input
- Each input corresponds to a square on the board
- If the inputted square is already occupied by another counter that value is treated as an invalid input and a relevant reinput prompt is printed
- After the player makes their move the board is reprinted with the move.
- The computer is then passed the turn
- The computer generates moves randomly, and in the case of an invalid generation a new move is generated. An invalid generation need not be shared with the user
- After the computer makes a valid move, the board is reprinted with the new move
- At any time, a winning state (4 of the same counter in an unbroken line) should end the game and communicate to the user who won and then return them to the main menu
- If there are no squares left to place a counter the game should end, the draw communicated to the user and then return to the main menu
- Selecting "Instructions" should print the instructions, accessed from a separate text file named "intructions.txt"
- The file should describe the programs functionality clearly to the user
- The instructions text file must be read line by line
- Selecting "Exit" should exit the program
- The "Exit" option must not rely on subroutines or functions

Gantt chart


Project name
Tic Tac Toe Project
Duration
60
Start date
14-Jun
End date
12-Aug
= Weekend
Task ID
Task Description
Task Duration
Start date
End date
1 Milestone 1 15 14-Jun 29-Jun
2 Defining and understanding the problem 15 14-Jun 29-Jun
3 Time management plan writup 1 20-Jun 21-Jun
4 Create Gantt chart 3 14-Jun 17-Jun
5 Program description/overview 3 14-Jun 17-Jun
6 Functional requirments 4 20-Jun 24-Jun
7 Context diagram 1 22-Jun 23-Jun
8 Milestone 2 36 30-Jun 5-Aug
9 Planning and designing software solutions 16 30-Jun 16-Jul
10 System models 2 30-Jun 2-Jul
11 Dataflow diagram 1 4-Jul 5-Jul
12 Structure diagram 1 6-Jul 7-Jul
13 IPO chart for subprogram 0 8-Jul 8-Jul
14 Screen design 2 30-Jun 2-Jul
15 Storyboard 2 30-Jun 2-Jul
16 Complete pseudocode 5 11-Jul 16-Jul
17 Flow chart for subprogram 0 8-Jul 8-Jul
18 Implementing software solutions 19 17-Jul 5-Aug
19 Coding solution 13 18-Jul 31-Jul
20 Data dictonary 4 1-Aug 5-Aug
21 User documentation 4 1-Aug 5-Aug
22 Milestone 3 6 6-Aug 12-Aug
23 Testing and evaluating data 3 6-Aug 9-Aug
24 Table of test data 1 6-Aug 7-Aug
25 Determine met requirements 1 8-Aug 9-Aug
26 Evaluate solution 1 8-Aug 9-Aug
27 Maintaining software solutions 2 10-Aug 12-Aug
28 Non-developer testing 0 10-Aug 10-Aug
29 Explain future maintenance development 2 11-Aug 13-Aug
30 Logbook 59 14-Jun 12-Aug
14/06/22
15/06/22
16/06/22
17/06/22
18/06/22
19/06/22
20/06/22
21/06/22
22/06/22
23/06/22
24/06/22
25/06/22
26/06/22
27/06/22
28/06/22
29/06/22
30/06/22
1/07/22
2/07/22
3/07/22
4/07/22
5/07/22
6/07/22
7/07/22
8/07/22
9/07/22
10/07/22
11/07/22
12/07/22
13/07/22
14/07/22
15/07/22
16/07/22
17/07/22
18/07/22
19/07/22
20/07/22
21/07/22
22/07/22
23/07/22
24/07/22
25/07/22
26/07/22
27/07/22
28/07/22
29/07/22
30/07/22
31/07/22
1/08/22
2/08/22
3/08/22
4/08/22
5/08/22
6/08/22
7/08/22
8/08/22
9/08/22
10/08/22
11/08/22
12/08/22

Time Management Plan

One of the most difficult factors of multistep projects such as this one is efficient management and use of time. In order to successfully create a software solution that fulfills the requirements, the Software Development Cycle will be used. This will be spread across the nearly 3-month project timespan. The Gantt chart above is essential in planning out dependencies, order and the time needed for each individual component. A logbook will be kept to allow clarity on how the project is progressing and keeping to the routine set forth by the initial time management strategy. This will also allow for reflection on difficulties encountered and what still needs to be completed to focus later work sessions. This breaking down of urgent tasks and focused work will maximize the time efficiency of the project. A version control will be kept in cloud storage. In the case of errors, it will be far easier to revert to a pre-error version, and there will backups of project relevant files in the case of file loss. The version control can save huge amounts of time in these scenarios. It also acts as a verification process for the logbook, that is every logbook entry can be checked against the relevant version control files to detect errors and inconsistencies. This increases the accuracy of the logbook, and thus its efficacy. Each of these components helps manage the development cycle of this project, and complete work efficiently in a useful manner.

Context Diagram