|  |  |
| --- | --- |
| Logo  Description automatically generated | **AIR UNIVERSITY** |
| **DEPARTMENT OF COMPUTER SCIENCE** |
| **PROJECT REPORT** |

**Student Name: Hamza Umer Farooq Reg. No: 200789**

**Subject: OPERATING SYSTEMS Semester: IV**

**Objective: LUDO GAME IN C++ USING Operating Systems Concepts**

**ASSESSMENT:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attributes** | **Excellent**  **(5)** | **Good**  **(4)** | **Average**  **(3)** | **Satisfactory**  **(2)** | **Unsatisfactory (1)** |
| **Ability to Conduct**  Task |  |  |  |  |  |
| **Ability to assimilate the results** |  |  |  |  |  |
| **Effective use of theorems/postulates/formulas** |  |  |  |  |  |

Total Marks:

Obtained Marks:

**REPORT ASSESSMENT:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attributes** | **Excellent**  **(5)** | **Good**  **(4)** | **Average**  **(3)** | **Satisfactory**  **(2)** | **Unsatisfactory**  **(1)** |
| **Data presentation** |  |  |  |  |  |
| **Experimental results** |  |  |  |  |  |
| **Conclusion** |  |  |  |  |  |

**Ludo Board Game**

SYSTEM SPECIFICATIONS:

Hamza Umer Farooq – 200789 (TL)

CPU: AMD Ryzen 7 4700U  
GPU: AMD Radeon Vega (Integrated)  
RAM: 8GB DDR4  
OS used: Windows 11 with WSL 2.0 for Ubuntu 20.04

Sarib Hanif – 201947

CPU: Intel i5 1155g7  
GPU: Intel iris Xe (Integrated)  
RAM: 8GB DDR4  
OS used: Ubuntu 20.04 on VMWare

GROUP CONTRIBUTIONS:

Hamza Umer Farooq – 200789

1. PHASE 1 COMPLETE

+

1. Created Grid
2. Created Player and token Struct
3. Created Master Thread
4. Created Player Kill
5. Created Movement of Player on Grid
6. Threads and Semaphores Implementation

Sarib Hanif – 201947

1. Created Input of Number of Tokens/Player
2. Player Turns Randomization
3. Created player return if no hitrate
4. Created Disqualify, if no 6 in 10 turns
5. Created Safe Point to No Kill
6. Created Positioning System (1st, 2nd and so on)

S

Passing Parameters to each thread:

The parent thread will create the worker threads, passing each worker the location that it must check in the grid. This step will require passing several parameters to each thread.

A struct is made to create a data structure for each thread, which includes row and column where a thread must begin validating

struct token {  
 int value, x, y;  
 bool open, home, stop, win;  
 char sym;  
  
 token() {//CONSTRUCTORS  
 value = -1;  
 x = 0;  
 y = 0;  
 stop = 0;  
 win = 0;  
 open = 0;  
 sym = '!';  
 }  
  
 token(int v, int x1, int y1, bool o, bool s, bool w, char sy) {  
 value = v;  
 x = x1;  
 y = y1;  
 stop = s;  
 win = w;  
 open = o;  
 sym = sy;  
 }  
  
};

A player thread is made, which has the data for hits, turns with sixes, and scoreboard level

Designing the board:

i/j 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

1

2 | - - - |

3 & & | - B B | # #

4 | B B - |

5 | - B - |

6 & & | - B - | # #

7 --- --- --- --- --- --- - B - --- --- --- --- --- ---

8 - A - - - - H H H - - - D - -

9 - A A A A A H H H D D D D D -

10 - - A - - - H H H - - - - D -

11 --- --- --- --- --- --- - C - --- --- --- --- --- ---

12 % % | - C - | @ @

13 | - C - |

14 | - C C |

15 % % | C C - | @ @

| - - - |

Grid of 15\*15 Matrix

Pseudocode Phase 1:

1. Make a 15\*15 array
2. Place Tokens according to the number of tokens chosen (1/2/3/4)
3. Make rest of board using positions for 2D arrays.

Pseudocode Phase 2:

Complete Ludo Implementation

1. Tokens implemented
   1. Using semaphores
   2. Every player giver 1-4 threads according to number of tokens chosen
   3. Total threads therefore will be 4-16
2. Turn Randomization
   1. Turns will be given in an alternate fashion.
   2. As asked to give random every time. It makes the program sometimes go in starvation as some threads might never be called
   3. So only first time random is called in which the turns then repeat.
3. Safe squares
   1. Safe squares have been made on the board (total 8, NOT INCLUDING THE HOME PATH)
   2. On these safe points the player cannot be killed.
   3. On any other point on the table the individual thread is cancelled and sent back to home using 2D array position.
4. No hits
   1. If a player doesn’t hit another player, he cant go to HOME
   2. i.e they will continue to circle around the board till they have a hit
5. 3 Sixes
   1. If a player gets three sixes in a row, their turn will be cancelled and the dice will be passed on to the next player
6. Winners will be declared according to the ones who take all their tokens into home first and so on.