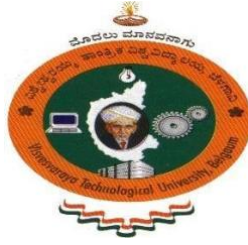


**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**  
**Jnana Sangama, Belagavi-590010**



**MINI PROJECT REPORT**  
**ON**  
**“2048 USING C”**

Submitted in partial fulfillment for the requirements for the seventh semester curriculum

**BACHELOR OF ENGINEERING**  
**IN**  
**COMPUTER SCIENCE AND ENGINEERING**

For the Academic year 2020-2021

Submitted by:

**PIYUSHRAJ SINGH**  
**PRAGYA RAVINDRAKUMAR MUGALE**

**1MV20CS075**  
**1MV20CS077**

Project carried out at:

**Sir M. Visvesvaraya Institute of Technology**  
Bengaluru-562157

Under the guidance of:

**Mrs. NEETHU**

Assistant Professor, Department of CSE  
Sir M. Visvesvaraya Institute of Technology, Bengaluru



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**SIR. M VISVESVARAYA INSTITUTE OF TECHNOLOGY**  
HUNASAMARANAHALLI, BENGALURU-562157

## TABLE OF CONTENTS

<b>S.NO</b>	<b>CONTENT</b>	<b>PAGE NO</b>
1	DECLARATION	1
2	ABSTRACT	2
3	INTRODUCTION	3
4	SYSTEM REQUIREMENT	4
5	SYSTEM ANALYSIS	5
6	SYSTEM DESIGN	7
7	PROJECT IMPLEMENTATION	8
8	TESTING	21
9	CONCLUSION	25
10	BIBLIOGRAPHY	26

## **DECLARATION**

We hereby declare that the entire mini project work embodied in this dissertation has been carried out by us and no part has been submitted for any degree or diploma of any institution previously.

Place: Bengaluru

Date:

Signature of Students:

**PIYUSHRAJ SINGH**  
(1MV20CS075)

**PRAGYA RAVINDRAKUMAR MUGALE**  
(1MV20CS077)

## **ABSTRACT**

2048 is a single-player sliding the puzzle video game. The objective of the game is to slide numbered tiles on a grid to combine them to create a tile with the number 2048; however, one can continue to play the game after reaching the goal, creating tiles with larger numbers.

# **INTRODUCTION**

## **INTRODUCTION TO C**

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language. It keeps fluctuating at number one scale of popularity along with Java programming language, which is also equally popular and most widely used among modern software programmers.

## **PROBLEM STATEMENT**

Construction of 2048 game using C

## **SYSTEM REQUIREMENT**

### **HARDWARE REQUIREMENT**

Processor :- 11th Gen Intel(R) Core(TM) i5-11300H @ 3.1 GHz

RAM :- 8GB

System Type :- 64-bit operating system, x64 based processor

Operating System :- Windows 11 Home

### **SOFTWARE REQUIREMENT**

IDE :- Visual Studio Code

Compiler :- minGW

# **SYSTEM ANALYSIS**

## **INTRODUCTION TO SYSTEM ANALYSIS**

System analysis is the process of observing systems for troubleshooting or development purposes. It is applied to information technology, where computer-based systems require defined analysis according to their makeup and design.

System analysis can include looking at end-user implementation of a software package or product; looking in-depth at source code to define the methodologies used in building software; or taking feasibility studies and other types of research to support the use and production of a software product, among other things.

## **EXISTING SYSTEM**

At any point, there are only four possible moves that we can make. Sometimes, some of these moves have no impact on the board and are thus not worth making – for example, in the above board a move of “Down” will have no impact since all of the tiles are already on the bottom edge.

The challenge is then to determine which of these four moves is going to be the one that has the best long-term outcome.

Our algorithm is based on the Expectimax algorithm, which is itself a variation of the Minimax algorithm, but where the possible routes through our tree are weighted by the probability that they will happen.

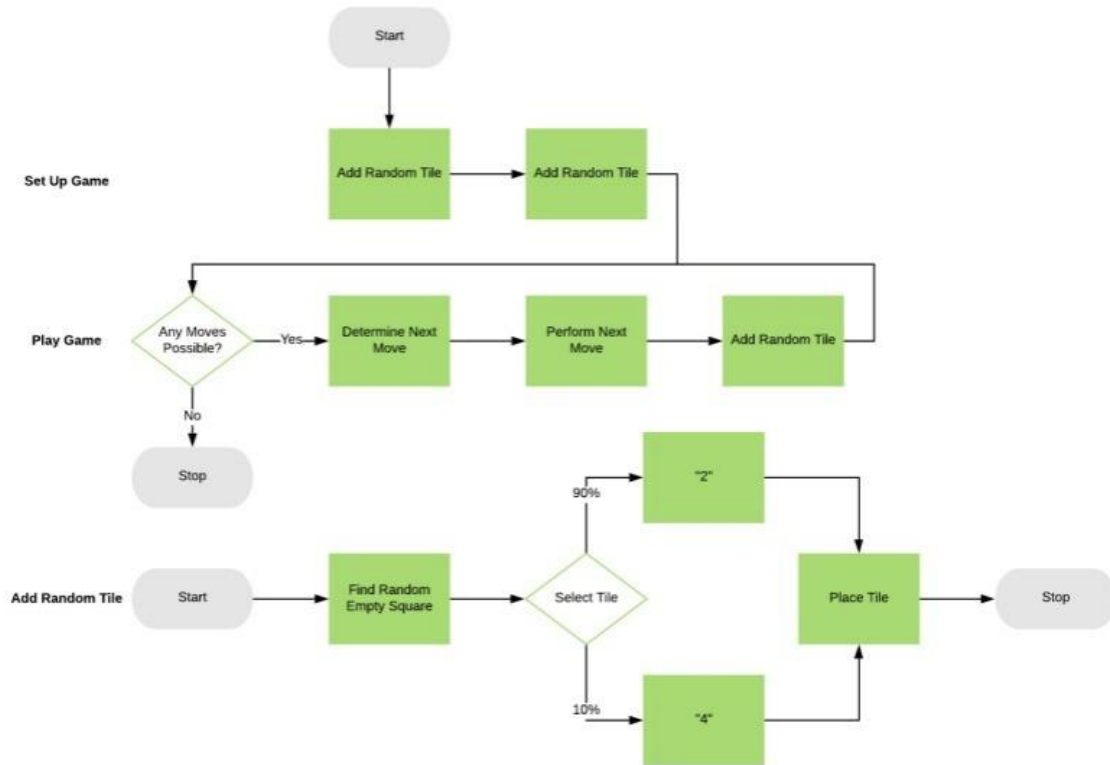
Essentially, we treat the game as a two-player game:

Player One – the human player – can shift the board in one of four directions

Player Two – the computer player – can place a tile into an empty location on the board

Based on this, we can generate a tree of outcomes from each move, weighted by the probability of each move happening. This can then give us the details needed to determine which human move is likely to give the best outcome.

---



## PROPOSED SYSTEM

A small upgrade would be, to make it a two-player game where the second player decides where the random number would be put in order to stop the first player from reaching 2048.



# SYSTEM DESIGN

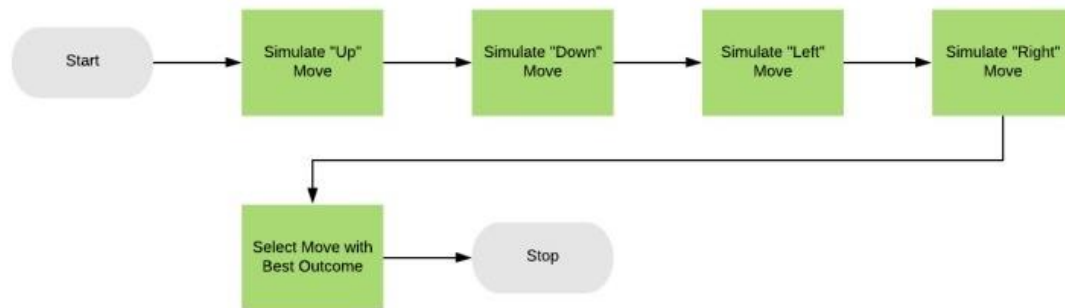
## SYSTEM ARCHITECTURE

Solving this game is an interesting problem because it has a random component. It's impossible to correctly predict not only where each new tile will be placed, but whether it will be a "2" or a "4".

As such, it is impossible to have an algorithm that will correctly solve the puzzle every time. The best that we can do is determine what is likely to be the best move at each stage and play the probability game.

So we have now reduced our algorithm into simulating any given move and generating some score for the outcome.

This is a two-part process. The first pass is to see if the move is even possible, and if not, then abort early with a score of "0". If the move is possible, then we'll move on to the real algorithm where we determine how good a move this is.



## PROJECT IMPLEMENTATION

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <conio.h>
#include <ctype.h>
#include <windows.h>
#include <process.h>
```

```
#define UP 72
#define DOWN 80
#define LEFT 75
#define RIGHT 77
#define SIZE 4
#define BORDER 219
#define ESC 57
#define PR 2
```

```
int Matrix[SIZE+1][SIZE+1]={0};
int score=2048,last=0;
```

```
void starting();
void wait();
void waitL();
void vname();
void Display();
void print_ever();
void Action(int);
void Random_creator();
void Starting_Random();
int Temp_counter();
```

```
void ending();
void record();
void Down();
void Up();
void Left();
void Right();
```

```
void starting()
{
    printf("=====2048
GAME=====");
    printf("\n\n\n\t\tINSTRUCTION\n\n\n");
    printf(" -> Enter arrow key to move\n\n");
    printf(" -> For wining this game any one box have value 2048\n\n");
    printf(" -> You have maximum 2048 try to win the game\n\n\n\n");
    printf("\t\t\t\t\tPRESS ANY KEY TO PLAY\n");
    while(!kbhit());
}
```

```
void wait()
{
    int i;
    for(i=0;i<4500000;i++);
}
```

```
printer(char s[])
{
    int i=0;
    while(s[i]!='\0')
    {
```

```
        printf("%c",s[i]);

        i++;
    }
}

void Action(int Arrow)
{

    switch(Arrow)
    {
        case UP:
            {
                Up();
                break;
            }
        case DOWN:
            {
                /* logic here */
                Down();
                break;
            }
        case LEFT:
            {
                Left();
                break;
            }
        case RIGHT:
            {
                Right();
                break;
            }
    }
}
```

```
    default:
    {
        /*DO NOTHING */
        return;
    }
}
score--;
Random_creator();
Display();
}
```

```
void Random_creator()
{
    int temp1,temp2,add,i,j;
    srand(time(NULL));
    temp1=rand()%SIZE; // i

    srand(time(NULL));
    temp2=rand()%SIZE; // j

    if((temp1+temp2)%2==0)
        add=2;
    else
        add=4;

    for(i=0;i<temp1;i++)
    {
        for(j=temp2;j<SIZE;j++)
        {
            if(Matrix[i][j]==0)
            {
                Matrix[i][j]=add;
            }
        }
    }
}
```



```

}

void print_ever()
{
    printf("\n\n\n\t\t2048\n");
    printf("\t\t\t\t\t SCORE : %d\n\n\n",score);
    printf("\t\t\t\t\t 
%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c\
n",BORDER,BORDER,BORDER,BORDER,BORDER,BORDER,BORDER,BOR
DER,BORDER,BORDER,BORDER,BORDER,BORDER,BORDER,BORDER,B
ORDER,BORDER,BORDER,BORDER,BORDER,BORDER,BORDER,BORDER
,BORDER,BORDER,BORDER);

}


void ending()
{
    /*here print prev high score and current score from file */
    printf("\n\n\t\t\tYOUR SCORE is : %d",score);
    record();
}

void record(){
    char plname[20],nplname[20],cha,c;
    int i,j;
    FILE *info;
    info=fopen("record.txt","a+");
    getch();
    system("cls");
    printf("Enter your name\n");
    scanf("%[^\\n]",plname);
    //*****

    for(j=0;plname[j]!='\\0';j++){ //to convert the first letter after space to capital
        nplname[0]=toupper(plname[0]);

```

```

if(plname[j-1]== ' '){
nplname[j]=toupper(plname[j]);
nplname[j-1]=plname[j-1];}
else nplname[j]=plname[j];
}
nplname[j]='\0';
//*****

//sdprintf(info,"\t\t\tPlayers List\n");
fprintf(info,"Player Name :%s\n",nplname);
//for date and time

time_t mytime;
mytime = time(NULL);
fprintf(info,"Played Date:%s",ctime(&mytime));
//*****

fprintf(info,"Score:%d\n",score);//call score to display score
//fprintf(info,"\nLevel:%d\n",10);//call level to display level
for(i=0;i<=50;i++)
fprintf(info,"%c",'_');
fprintf(info,"\n");
fclose(info);
printf("wanna see past records press 'y'\n");
cha=getch();
system("cls");
if(ch=='y'){
info=fopen("record.txt","r");
do{
    putchar(c=getc(info));
    }while(c!=EOF);}
fclose(info);
printf("\n\n\n\t\t\tPRESS ANY KEY TO EXIT\n");
while(!kbhit());

```



```
    system("attrib +h +s record.txt");  
}
```

```
void Starting_Random()  
{  
    Matrix[3][1]=8;  
    Matrix[3][2]=32;  
    Matrix[3][3]=16;  
    Matrix[2][2]=8;  
    Matrix[2][3]=8;  
    Matrix[1][2]=8;  
  
    Display();  
}
```

```
int Temp_counter()  
{  
    /* IT SHOULD FIND MAX VALUE FROM WHOLE MATRIX */  
    int temp=0,i,j;  
  
    for(i=0;i<SIZE;i++)  
    {  
        for(j=0;j<SIZE;j++)  
        {  
            if(Matrix[i][j]==2048)  
                return(1);  
            if(Matrix[i][j]==0)  
                temp=1;  
        }  
    }  
    if(temp==1)  
    {
```

```

        last=0;
        return(-99);
    }
    else
    {
        if(last==1)
        {
            return(0);
        }
        last=1;
        return(-99);
    }
}

```

```

void Down()
{
    int i,j;
    for(j=0;j<SIZE;j++)
    {
        i=2;
        while(1)
        {
            while(Matrix[i][j]==0)
            {
                if(i==-1)
                    break;
                i--;
            }
            if(i==-1)
                break;
            while(i<SIZE-1 && (Matrix[i+1][j]==0 || Matrix[i][j]==Matrix[i+1][j]))
            {

```

```

        Matrix[i+1][j]+=Matrix[i][j];
        Matrix[i][j]=0;
        i++;
    }
    i--;
}
}
}
void Up()
{
    int i,j;
    for(j=0;j<SIZE;j++)
    {
        i=1;
        while(1)
        {
            while(Matrix[i][j]==0)
            {
                if(i==SIZE)
                    break;
                i++;
            }
            if(i==SIZE)
                break;
            while(i>0 && (Matrix[i-1][j]==0 || Matrix[i][j]==Matrix[i-1][j]))
            {
                Matrix[i-1][j]+=Matrix[i][j];
                Matrix[i][j]=0;
                i--;
            }
            i++;
        }
    }
}

```

```

    }
}
void Right()
{
    int i,j;
    for(i=0;i<SIZE;i++)
    {
        j=2;
        while(1)
        {
            while(Matrix[i][j]==0)
            {
                if(j==-1)
                    break;
                j--;
            }
            if(j==-1)
                break;
            while(j<SIZE-1 && (Matrix[i][j+1]==0 || Matrix[i][j]==Matrix[i][j+1]))
            {
                Matrix[i][j+1]+=Matrix[i][j];
                Matrix[i][j]=0;
                j++;
            }
            j--;
        }
    }
}
void Left()
{
    int i,j;
    for(i=0;i<SIZE;i++)

```

```

{
    j=1;
    while(1)
    {
        while(Matrix[i][j]==0)
        {
            if(j==SIZE)
                break;
            j++;
        }
        if(j==SIZE)
            break;
        while(j>0 && (Matrix[i][j-1]==0 || Matrix[i][j]==Matrix[i][j-1]))
        {
            Matrix[i][j-1]+=Matrix[i][j];
            Matrix[i][j]=0;
            j--;
        }
        j++;
    }
}

main()
{
    int aro;
    char Arrow;
    char s[]="THANKS FOR PLAYING, GOOD LUCK FOR NEXT TIME ";
    int temp;
    starting();
    Starting_Random();
    Arrow=DOWN;
    while(Arrow!=ESC) // HERE IF USER WANT TO EXIT THEN PRESS ESC

```

KEY

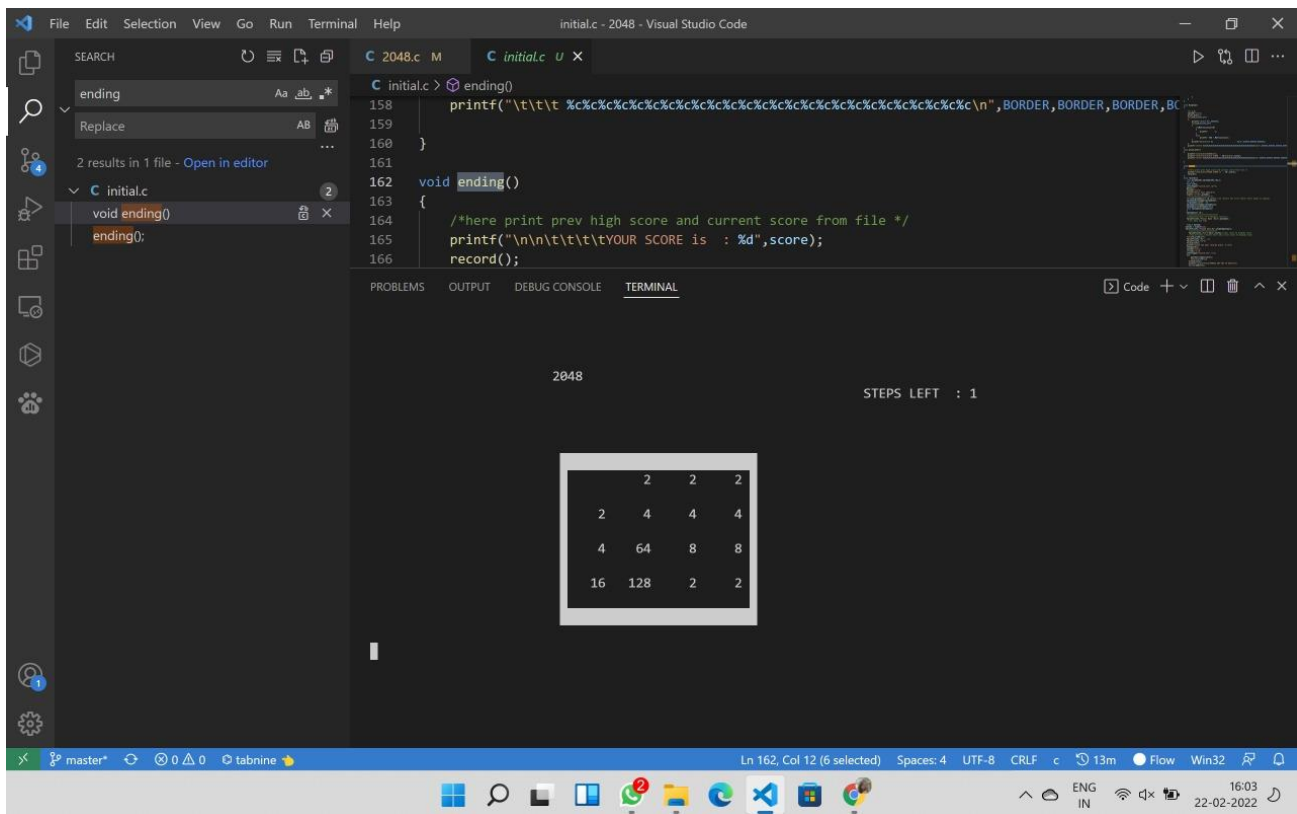
```
{
    Arrow=getch();
    aro=Arrow;
    Action(Arrow);
    temp=Temp_counter();
    if(temp==1)
    {
        printf("\n\t\tYOU WON");
        ending();
        break;
    }
    if(temp==0 || score<0)
    {
        printf("\n\t\tSORRY ! GAME OVER\n");
        break;
    }
}

system("cls");
printf("\n\n\n\t");
printer(s);
}
```

# TESTING

## DEFINITION OF TESTING

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is defect free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.



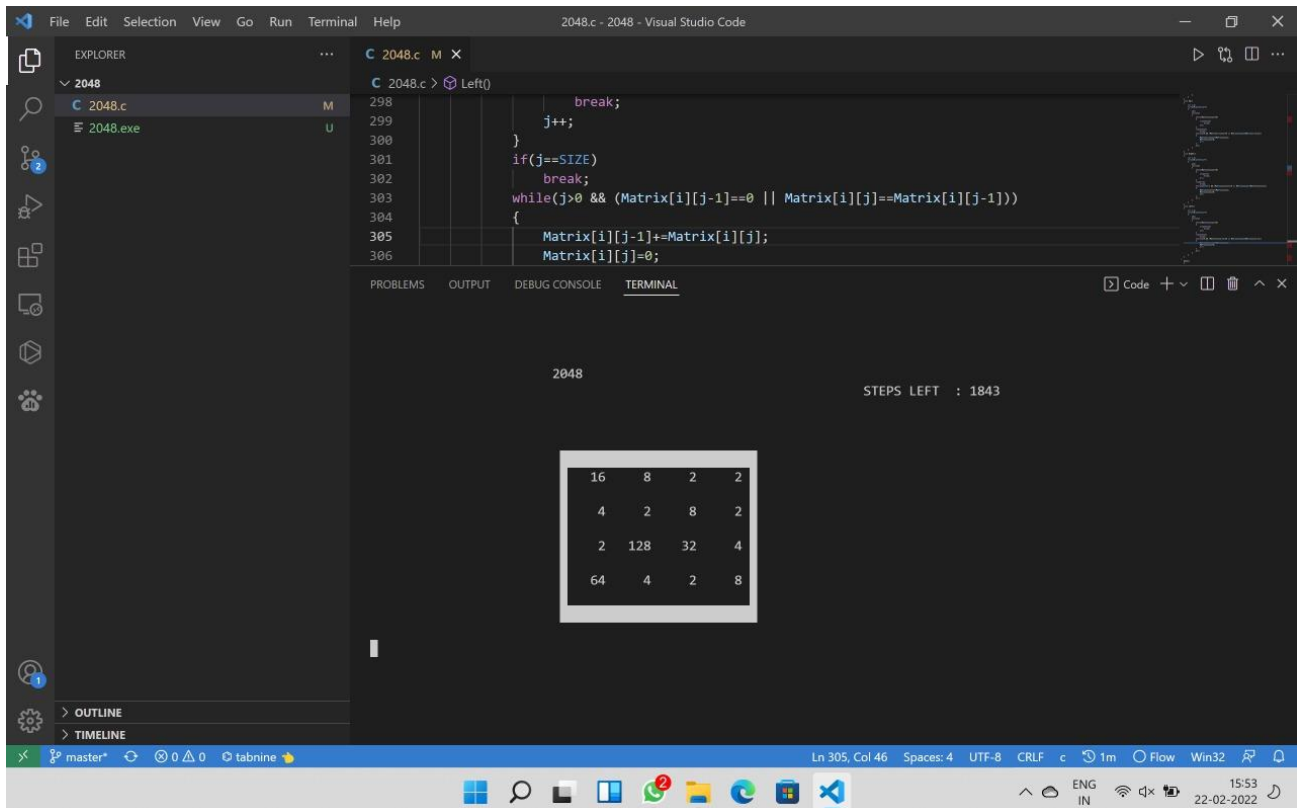
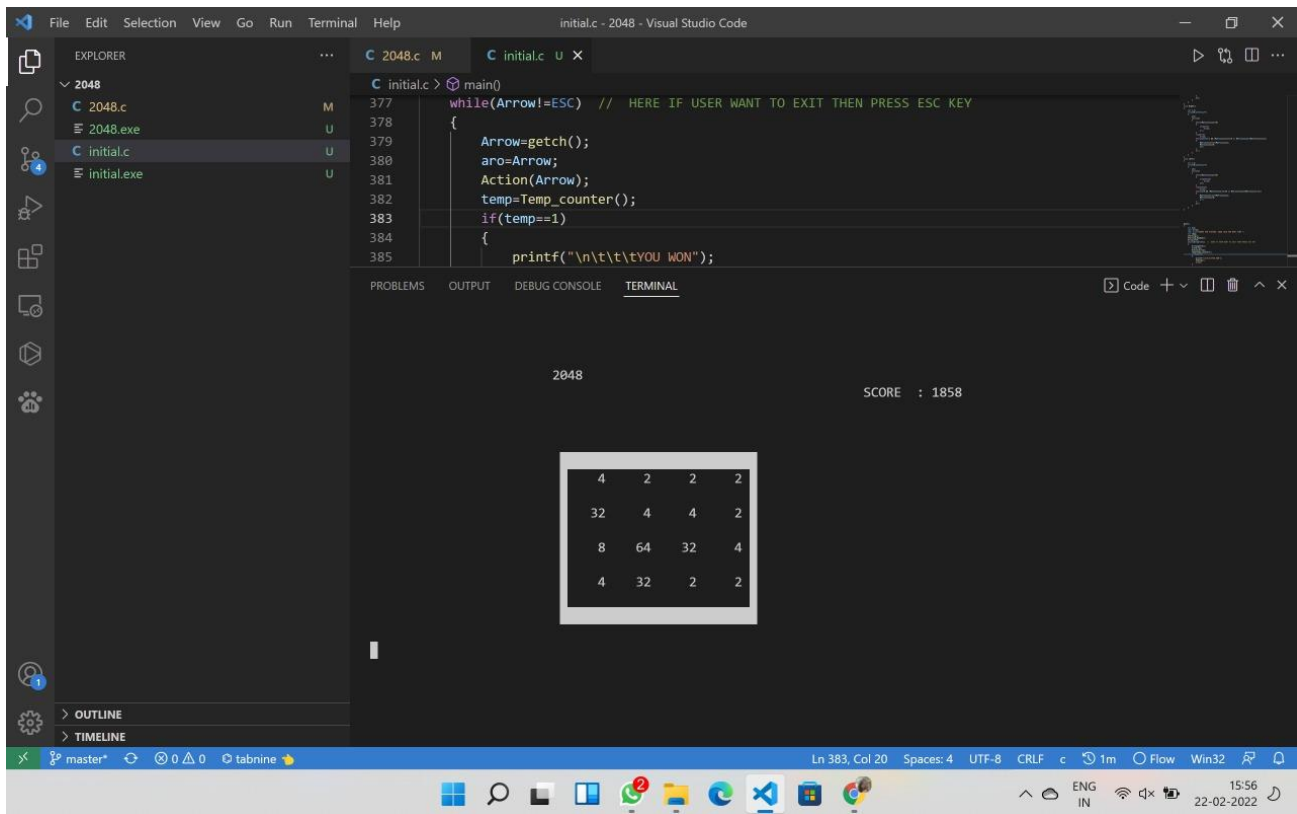
The screenshot shows the Visual Studio Code editor with a file named 'initial.c' open. The search bar on the left shows 'ending' with 2 results. The code in the editor includes a printf statement with a border pattern and a function 'void ending()' that prints a 4x4 grid of numbers. The terminal output shows the number '2048' and 'STEPS LEFT : 1' above the grid.

```
158 printf("\t\t\t %c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c\n",BORDER,BORDER,BORDER,BC
159 }
160
161
162 void ending()
163 {
164     /*here print prev high score and current score from file */
165     printf("\n\n\t\t\tYOUR SCORE is : %d",score);
166     record();
```

2048

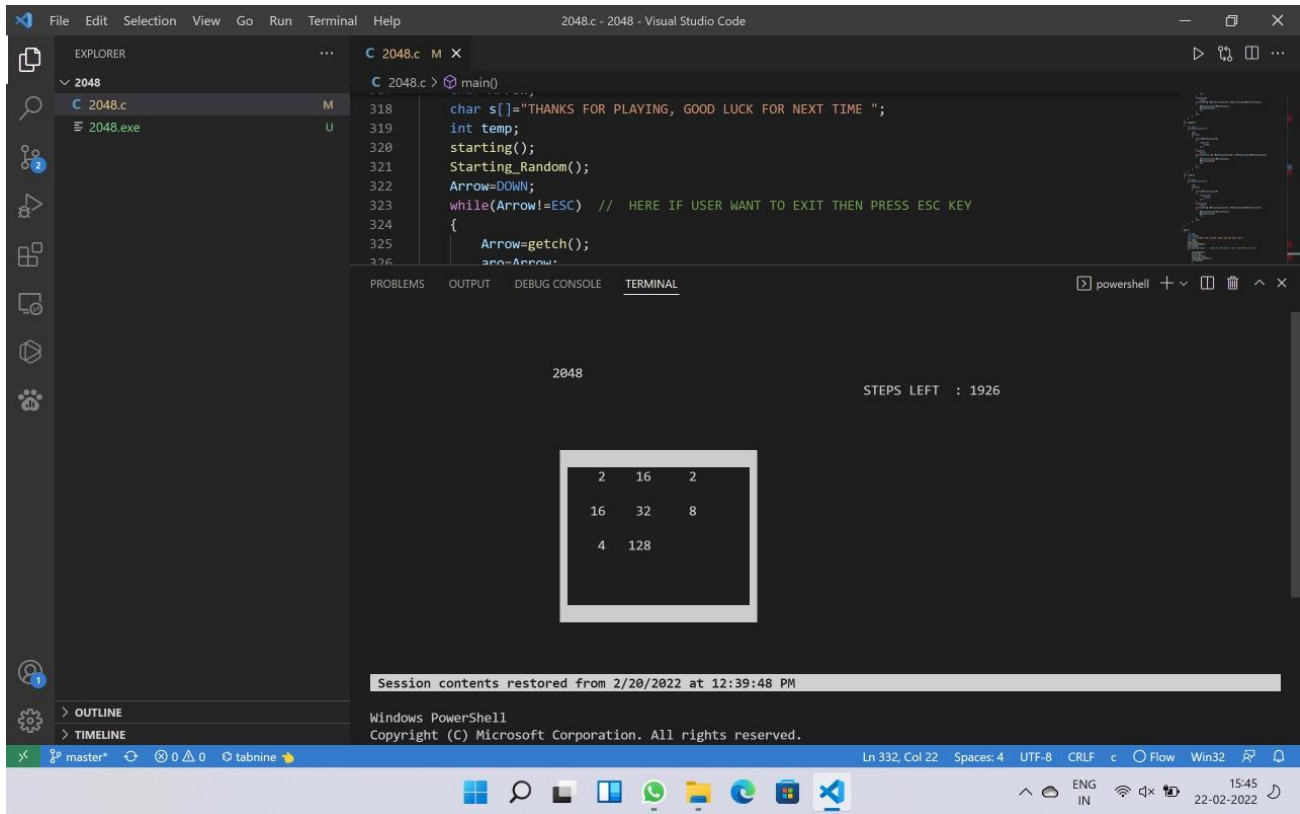
STEPS LEFT : 1

2	2	2	
2	4	4	4
4	64	8	8
16	128	2	2









## CONCLUSION

2048 is a highly interesting game to attempt to solve. There is no perfect way to solve it, but we can write heuristics that will search for the best possible routes through the game.

## **BIBLIOGRAPHY**

<https://stackoverflow.com/questions/22342854/what-is-the-optimal-algorithm-for-the-game-2048>

<https://code-projects.org/2048-game-in-c-with-source-code/>