**2018-2019**

|  |
| --- |
| **St Coloumbas’ School |** **C++ project** |

|  |  |
| --- | --- |
| RUCHIR JAIN | **MINESWEEPER** |

**18 XII-E**

Certificate

*This is to certify that Ruchir Jain of St.Columbas School, Class 12th has completed this project ‘ MINESWEEPER ’ under my supervision, and completed to my satisfaction.*

**Signature**

**Index**

|  |  |  |
| --- | --- | --- |
| S. No. | Content | Page No. |
| 1 | Aim | 4 |
| 2 | Header Files | 5 |
| 3 | User Defined Class | 6 |
| 4 | Source Code | 7 |
| 5 | Output Screen | 28 |
| 6 | Acknowledgement | 35 |

*Aim*

***When the world seems imperfect, the minefield reminds me that we are guided through existence by rigid, infallible rules. It is flawless in that way. Through my years of sweeping, I have come to realize that minesweeper has a lot of things to teach us in life. Here are some of these-***

***There is no trick or gimmick to successfully getting started.****In minesweeper, you begin games by randomly clicking to find openings from which you can work. There is no strategy for success in the beginning.*

***While situations may seem to have purpose or design, they do not.****Designs or patterns in the mines are purely coincidental and are not indicative of an overarching design or purpose in the mine field.*

***It sharpens our thinking speed and capacity.*** *It also improves our ability to solve simple problems that further helps us in studying.*

*Global functions used*

***Header Files***

|  |  |  |
| --- | --- | --- |
| Header files | Functions used | Used to |
| fstream.h | cin() | get input |
| cout() | get output |
| endl | inserts a new-line character and [flushes](http://www.cplusplus.com/basic_ostream::flush) the stream |
| open() | open file |
| close() | close file |
| stdio.h | gets() | reads characters from the standard input and stores them |
| dos.h | delay() | delay the output |
| time.h | clock() | returns the number of clock ticks elapsed since the program was launched |
| stringh.h | strcpy | copy 1 string to another |
| conio.h | gotoxy() | goto specific location |
| getch() | get a character from user |
| clrscr() | clear the screen |
| txtbackground() | set background colour |
| textcolor() | set text color |
| cprintf() | print on screen |
| kbhit | to determine if a key has been pressed or not |
| \_setcursortype() | set the cursors shape |
| graphics.h | getx | get x coordinate |
| gety | get y coordinate |
| stdlib.h | random() | generate random no |
| exit() | terminates the process normally |

**User Defined Classes**

**CLASS SCORE**

class score

{

char name[50];

int times,timem;

public:

score()

{

strcpy(name,"N/A");

times=0;

timem=0;

}

void accept(char[],int,int);

void display(int);

};

void score::accept(char n[50],int m,int s)

{

strcpy(name,n);

timem=m;

times=s;

};

void score::display(int i)

{

gotoxy(24,7+i);

cout<<name;

gotoxy(51,7+i);

cout<<timem<<":"<<times;

}

***Source Code***

#include<fstream.h>

#include<stdio.h>

#include<dos.h>

#include<time.h>

#include<string.h>

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#include<stdlib.h>

int minefield[20][20];

int field\_info[20][20];

int FIELD\_SIZE = 6;

int FIELD\_MINE\_NO = 3;

int OFFSET\_X = 25;

int OFFSET\_Y = 2;

const int FIELD\_MARKED = -2;

const int FIELD\_UNMARKED = -1;

const int MINE\_NOT\_PRESENT = 0;

const int MINE\_PRESENT = 1;

const int FIELD\_CLEARED = -4;

const int FIELD\_QUEUED = -3;

const int RESULT\_EXIT = -1;

const int RESULT\_WIN = 1;

const int RESULT\_EXPLOSION = 2;

const char FIELD\_CHAR = 219;

const char ZERO\_MINES = ' ';

const char MINE = '‑';

const char FIELD\_SELECT = 'S';

const char MINE\_EXPLOSION[] = "BOOM!!!! MINE EXPLODED!!!";

const char GAME\_WON[] = "Congrats !! You won !!";

const char GAME\_EXIT[] = "Thanks for playing. Press any key to continue ....";

const char NEVER\_TO\_BE\_PRINTED[] = "Abnormal result. Program AI crashed. You have exploded our minds. Congrats.";

const char WELCOME[] = " Welcome to Minesweeper ";

const char INSTRUCT[] = "Instructions :";

const char INSTRUCT\_MOVE[] = "Use 'w','a','s','d' to move selector";

const char INSTRUCT\_CLEAR[] = "Use 't' to check a space for mine";

const char INSTRUCT\_MINE[] = "Use 'm' to mark a space as possible mine";

const char INSTRUCT\_GIVEUP[] = "Use 'p' to get solution";

const char DIFFICULTY[] = "Please select a difficulty :";

const char DIFFICULTY\_1[] = "1. 6x6 , 9 mines";

const char DIFFICULTY\_2[] = "2. 10x10 , 20 mines";

const char DIFFICULTY\_3[] = "3. 20x20 , 40 mines";

const char CONTINUE[] = "Press any key to continue ...";

class score

{

char name[50];

int times,timem;

public:

score()

{

strcpy(name,"N/A");

times=0;

timem=0;

}

void accept(char[],int,int);

void display(int);

};

void score::accept(char n[50],int m,int s)

{

strcpy(name,n);

timem=m;

times=s;

};

void score::display(int i)

{

gotoxy(24,7+i);

cout<<name;

gotoxy(51,7+i);

cout<<timem<<":"<<times;

}

void winner(int,char);

void bfaccept(char[],int,int,char);

void scoreborder();

void printxy(char c,int x,int y)

{

int xi=getx(),yi=gety();

gotoxy(x+OFFSET\_X,y+OFFSET\_Y);

cout<<c;

gotoxy(xi,yi);

}

void print\_centre(const char s[],int y)

{

int len,j;

for(len=0;s[len]!='\0';len++);

gotoxy((80-len)/2,y);

cout<<s;

}

char initialize()

{

print\_centre(DIFFICULTY,8);

print\_centre(DIFFICULTY\_1,10);

print\_centre(DIFFICULTY\_2,12);

print\_centre(DIFFICULTY\_3,14);

char t;

while(1)

{

t = getch();

if(t == '1')

{

FIELD\_SIZE = 6;

FIELD\_MINE\_NO = 9;

break;

}

else if(t == '2')

{

FIELD\_SIZE = 10;

FIELD\_MINE\_NO = 20;

break;

}

else if(t == '3')

{

FIELD\_SIZE = 20;

FIELD\_MINE\_NO = 40;

break;

}

}

OFFSET\_X = (80-FIELD\_SIZE)/2;

OFFSET\_Y = 2;

clrscr();

int i,j,k;

for(i=0;i<FIELD\_SIZE;i++)

for(j=0;j<FIELD\_SIZE;j++)

{

minefield[i][j]= MINE\_NOT\_PRESENT;

field\_info[i][j]= FIELD\_UNMARKED;

}

for(k=0;k<FIELD\_MINE\_NO;k++)

{

int seed = random(FIELD\_SIZE\*FIELD\_SIZE);

i = seed/FIELD\_SIZE;

j = seed%FIELD\_SIZE;

if(minefield[i][j]== MINE\_PRESENT)

k--;

else

minefield[i][j]= MINE\_PRESENT;

}

for(i=0;i<FIELD\_SIZE;i++)

{

for(j=0;j<FIELD\_SIZE;j++)

printxy(FIELD\_CHAR,j+1,i+1);

cout<<endl;

}

return t;

}

void clear(int current\_x,int current\_y)

{

if(current\_x<1 || current\_x > FIELD\_SIZE || current\_y<1 || current\_y>FIELD\_SIZE)

return;

int totalmines = 0;

int upl = 1,upp = 1,upr = 1,right = 1,left = 1,dwnl = 1,down = 1,dwnr = 1;

if(current\_x == 1)

{

upl = left = dwnl = 0;

}

if(current\_y == 1)

{

upl = upp = upr = 0;

}

if(current\_x == FIELD\_SIZE)

{

upr = right = dwnr = 0;

}

if(current\_y == FIELD\_SIZE)

{

dwnl = down = dwnr = 0;

}

if(upl)

totalmines += minefield[current\_y-2][current\_x-2];

if(upp)

totalmines += minefield[current\_y-2][current\_x-1];

if(upr)

totalmines += minefield[current\_y-2][current\_x];

if(right)

totalmines += minefield[current\_y-1][current\_x];

if(left)

totalmines += minefield[current\_y-1][current\_x-2];

if(dwnl)

totalmines += minefield[current\_y][current\_x-2];

if(down)

totalmines += minefield[current\_y][current\_x-1];

if(dwnr)

totalmines += minefield[current\_y][current\_x];

if(totalmines==0)

printxy(ZERO\_MINES,current\_x,current\_y);

else

printxy('0'+totalmines,current\_x,current\_y);

field\_info[current\_y-1][current\_x-1]=totalmines;

if(totalmines == 0)

{

field\_info[current\_y-1][current\_x-1] = FIELD\_CLEARED;

if(upl && field\_info[current\_y-2][current\_x-2]!= FIELD\_CLEARED)

field\_info[current\_y-2][current\_x-2] = FIELD\_QUEUED;

if(upp && field\_info[current\_y-2][current\_x-1]!= FIELD\_CLEARED)

field\_info[current\_y-2][current\_x-1] = FIELD\_QUEUED;

if(upr && field\_info[current\_y-2][current\_x]!= FIELD\_CLEARED)

field\_info[current\_y-2][current\_x] = FIELD\_QUEUED;

if(right && field\_info[current\_y-1][current\_x]!= FIELD\_CLEARED)

field\_info[current\_y-1][current\_x] = FIELD\_QUEUED;

if(left && field\_info[current\_y-1][current\_x-2]!= FIELD\_CLEARED)

field\_info[current\_y-1][current\_x-2] = FIELD\_QUEUED;

if(dwnl && field\_info[current\_y][current\_x-2]!= FIELD\_CLEARED)

field\_info[current\_y][current\_x-2] = FIELD\_QUEUED;

if(down && field\_info[current\_y][current\_x-1]!= FIELD\_CLEARED)

field\_info[current\_y][current\_x-1] = FIELD\_QUEUED;

if(dwnr && field\_info[current\_y][current\_x]!= FIELD\_CLEARED)

field\_info[current\_y][current\_x] = FIELD\_QUEUED;

}

}

void clrqueue()

{

int y,x,clear\_queue=1;

while(clear\_queue!=0)

{

for(y=1;y<=FIELD\_SIZE;y++)

for(x=1;x<=FIELD\_SIZE;x++)

if(field\_info[y-1][x-1] == FIELD\_QUEUED)

clear(x,y);

clear\_queue = 0;

for(y=1;y<=FIELD\_SIZE;y++)

for(x=1;x<=FIELD\_SIZE;x++)

if(field\_info[y-1][x-1] == FIELD\_QUEUED)

clear\_queue++;

}

}

void show\_all()

{

int x,y,k;

for(y=1;y<=FIELD\_SIZE;y++)

for(x=1;x<=FIELD\_SIZE;x++)

{

if(minefield[y-1][x-1]== MINE\_PRESENT)

printxy(MINE,x,y);

else

clear(x,y);

}

}

int start()

{

gotoxy(1,1);

printxy(FIELD\_SELECT,1,1);

int exit = 0,i,j,k;

int current\_x = 1,current\_y = 1;

while(exit != 1)

{

char a=getch();

if(a == 'e')

{

exit = 1;

continue;

}

else if(a == 'p')

{

show\_all();

getch();

exit = 1;

continue;

}

else if(a=='w' || a=='s' || a=='d' || a=='a')

{

if(field\_info[current\_y-1][current\_x-1]== FIELD\_UNMARKED)

printxy(FIELD\_CHAR,current\_x,current\_y);

else if(field\_info[current\_y-1][current\_x-1] == FIELD\_MARKED)

printxy(MINE,current\_x,current\_y);

else if(field\_info[current\_y-1][current\_x-1] == FIELD\_CLEARED)

printxy(ZERO\_MINES,current\_x,current\_y);

else

printxy('0'+field\_info[current\_y-1][current\_x-1],current\_x,current\_y);

switch(a)

{

case 'w':if(current\_y == 1)

current\_y = FIELD\_SIZE;

else

current\_y--;

break;

case 'a':if(current\_x == 1)

current\_x = FIELD\_SIZE;

else

current\_x--;

break;

case 's':if(current\_y == FIELD\_SIZE)

current\_y = 1;

else

current\_y++;

break;

case 'd':if(current\_x == FIELD\_SIZE)

current\_x = 1;

else

current\_x++;

break;

}

printxy(FIELD\_SELECT,current\_x,current\_y);

}

else if(a=='t')

{

if(minefield[current\_y-1][current\_x-1]==MINE\_PRESENT)

return RESULT\_EXPLOSION;

else if(field\_info[current\_y-1][current\_x-1] == FIELD\_UNMARKED)

{

clear(current\_x,current\_y);

clrqueue();

printxy(FIELD\_SELECT,current\_x,current\_y);

}

int y,x,total = 0;

for(y=1;y<=FIELD\_SIZE;y++)

for(x=1;x<=FIELD\_SIZE;x++)

if(field\_info[y-1][x-1] == FIELD\_MARKED || field\_info[y-1][x-1] == FIELD\_UNMARKED)

total++;

if(total == FIELD\_MINE\_NO)

return RESULT\_WIN;

}

else if(a=='m' && field\_info[current\_y-1][current\_x-1] == FIELD\_UNMARKED)

{

field\_info[current\_y-1][current\_x-1] = FIELD\_MARKED;

printxy(MINE,current\_x,current\_y);

}

else if(a=='m' && field\_info[current\_y-1][current\_x-1] == FIELD\_MARKED)

{

field\_info[current\_y-1][current\_x-1] = FIELD\_UNMARKED;

printxy(FIELD\_CHAR,current\_x,current\_y);

}

}

return RESULT\_EXIT;

}

void border()

{

clrscr();

textbackground(BLACK);

clrscr();

gotoxy(1,1);

cout<<"Loaded MINESWEEPER v1.5 by Kuber Rawat";

textcolor(CYAN);

cout<<"\n";

for(int i=1;i<79;i++)

cprintf("\_");

for(i=1;i<23;i++)

{

gotoxy(1,i+1);

cprintf("||");

}

for(i=1;i<79;i++)

cprintf("\_");

for(i=1;i<23;i++)

{

gotoxy(79,i+1);

cprintf("||");

}

textcolor(BLUE);

}

void welcome()

{

while(!kbhit())

{

delay(100);

textbackground(BLACK);

textcolor(random(16));

gotoxy(20,4);

cprintf("ﾂｲﾂ ﾂｲﾂ ｲﾂｲﾂｲﾂｲ ﾂｲﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ");

gotoxy(20,5);

cprintf("ﾂｲﾂｲ ｲﾂｲﾂ ｲﾂｲ ﾂｲﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ ");

gotoxy(20,6);

cprintf("ﾂｲﾂｲT Tｲﾂｲﾂ ｲﾂｲ ﾂｲﾂｲﾂ ｲﾂ ｲﾂｲ ");

gotoxy(20,7);

cprintf("ﾂｲﾂ ﾂｲﾂ ﾂｲﾂ ｲﾂｲ ﾂｲﾂ ﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ ");

gotoxy(20,8);

cprintf("ﾂｲﾂ ﾂｲﾂ ﾂｲﾂ ｲﾂｲ ﾂｲﾂ ﾂｲﾂ ｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ ");

gotoxy(20,9);

cprintf("ﾂｲﾂ ﾂｲﾂ ﾂｲﾂ ｲﾂｲ ﾂｲﾂ ﾂｲﾂｲ ｲﾂ ｲﾂｲ ");

gotoxy(20,10);

cprintf("ﾂｲﾂ ﾂｲﾂ ﾂｲﾂ ｲﾂｲ ﾂｲﾂ ﾂｲﾂｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ ");

gotoxy(20,11);

cprintf("ﾂｲﾂ ﾂｲﾂ ﾂｲﾂ ｲﾂｲﾂｲﾂｲ ﾂｲﾂ ﾂｲﾂｲﾂ ｲﾂｲﾂｲﾂｲﾂｲ ");

gotoxy(10,14);

cprintf(" ｲﾂｲﾂｲﾂ ｲﾂｲ ｲ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂ ");

gotoxy(10,15);

cprintf("ｲﾂｲﾂ ｲﾂｲ ﾂｲﾂ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂｲ ");

gotoxy(10,16);

cprintf("ｲﾂｲﾂ ﾂｲ ﾂｲﾂ ｲﾂ ｲﾂｲ ｲﾂｲ ｲﾂ ｲﾂｲ ｲﾂｲ ｲﾂ ｲﾂｲ ");

gotoxy(10,17);

cprintf(" ﾂｲﾂｲﾂｲﾂ ﾂｲ ｲﾂｲﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂ ");

gotoxy(10,18);

cprintf(" ﾂｲﾂｲﾂ ﾂｲ ｲﾂｲﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂ ");

gotoxy(10,19);

cprintf(" ｲﾂｲﾂ ｲ ｲﾂｲﾂｲ ｲ ｲﾂｲ ｲﾂｲ ｲﾂｲﾂ ｲﾂｲ ｲﾂｲ ｲﾂ ");

gotoxy(10,20);

cprintf(" ﾂｲﾂｲ ｲ ｲﾂ ﾂｲ ｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂｲ ");

gotoxy(10,21);

cprintf("ｲﾂｲﾂｲﾂ ﾂｲ ｲﾂ ｲﾂｲﾂｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂｲ ｲﾂｲﾂｲﾂｲ ｲﾂ ｲﾂｲﾂ");

}

clrscr();

getch();

}

void instructions()

{

print\_centre(WELCOME,6);

print\_centre(INSTRUCT,8);

print\_centre(INSTRUCT\_MOVE,9);

print\_centre(INSTRUCT\_CLEAR,10);

print\_centre(INSTRUCT\_MINE,11);

print\_centre(INSTRUCT\_GIVEUP,12);

print\_centre(INSTRUCT,8);

print\_centre(CONTINUE,13);

getch();

}

void play()

{

char diff;

diff=initialize();

int result;

int time=clock();

result = start();

time=clock()-time;

clrscr();

if(result == RESULT\_EXPLOSION)

print\_centre(MINE\_EXPLOSION,3);

else if(result == RESULT\_EXIT)

print\_centre(GAME\_EXIT,3);

else if(result == RESULT\_WIN)

{

print\_centre(GAME\_WON,3);

winner(time,diff);

}

else

print\_centre(NEVER\_TO\_BE\_PRINTED,3);

getch();

\_setcursortype(\_NORMALCURSOR);

}

void winner(int t,char diff)

{

char n[50];

int i,s,m;

m=t/60;

s=t%60;

textcolor(CYAN);

print\_centre("HIGHSCORE",4);

gotoxy(15,6);

textcolor(GREEN);

for(i=0;i<25;i++)

cout<<"ｲｲ";

for(i=1;i<9;i++)

{

gotoxy(15,i+6);

cout<<"ｲｲ";

}

gotoxy(15,14);

for(i=0;i<25;i++)

cout<<"ｲｲ";

for(i=1;i<9;i++)

{

gotoxy(63,i+6);

cout<<"ｲｲ";

}

gotoxy(15,10);

for(i=0;i<24;i++)

cout<<"ｲｲ";

textcolor(WHITE);

gotoxy(30,12);

cout<<"TIME:"<<m<<" : "<<s;

gotoxy(30,8);

cout<<"NAME:";

gets(n);

bfaccept(n,m,s,diff);

}

void bfaccept(char n[50],int m,int s,char diff)

{

score sc;

ofstream file;

if(diff=='1')

file.open("EASY.dat",ios::binary|ios::app);

else if(diff=='2')

file.open("MEDIUM.dat",ios::binary|ios::app);

else

file.open("HARD.dat",ios::binary|ios::app);

sc.accept(n,m,s);

file.write((char\*)&sc,sizeof(sc));

file.close();

}

void bfprint(int diff)

{

int i=0;

score sc;

fstream file;

if(diff==1)

file.open("EASY.dat",ios::binary|ios::in);

else if(diff==2)

file.open("MEDIUM.dat",ios::binary|ios::in);

else

file.open("HARD.dat",ios::binary|ios::in);

while(!file.eof())

{

file.read((char\*)&sc,sizeof(sc));

if(file.eof())

break;

sc.display(i);

i=i+4;

if(i%16==0)

{

clrscr();

scoreborder();

i=0;

}

}

file.close();

}

void menuborder()

{

int i;

gotoxy(15,6);

textcolor(GREEN);

for(i=0;i<25;i++)//Upper border

cprintf("ｲｲ");

for(i=1;i<15;i++)//Left border

{

gotoxy(15,i+6);

cprintf("ｲｲ");

}

gotoxy(15,20);

for(i=0;i<25;i++)//Lower border

cprintf("ｲｲ");

for(i=1;i<15;i++)

{

gotoxy(63,i+6);

cprintf("ｲｲ");

}

gotoxy(15,16);

for(i=0;i<24;i++)

cprintf("ｲｲ");

textcolor(WHITE);

}

void scoreborder()

{

int i;

gotoxy(15,3);

textcolor(GREEN);

for(i=0;i<25;i++)//Upper border

cprintf("ｲｲ");

for(i=1;i<20;i++)//Left border

{

gotoxy(15,i+3);

cprintf("ｲｲ");

}

gotoxy(15,23);

for(i=0;i<25;i++)//Lower border

cprintf("ｲｲ");

for(i=1;i<20;i++)

{

gotoxy(63,i+3);

cprintf("ｲｲ");

}

textcolor(WHITE);

gotoxy(24,5);

cprintf("NAME");

gotoxy(51,5);

cprintf("TIME");

}

void main()

{

int ch,chh;

border();

welcome();

clrscr();

do

{

clrscr();

menuborder();

textcolor(WHITE);

print\_centre(" MENU ",6);

print\_centre("1. PLAY ",8);

print\_centre("2. INSTRUCTIONS",10);

print\_centre("3. HIGHSCORE ",12);

print\_centre("4. EXIT ",14);

print\_centre("Enter your Choice : ",18);

cin>>ch;

switch(ch)

{

case 1:clrscr();

menuborder();

play();

break;

case 2:clrscr();

menuborder();

instructions();

break;

case 3:clrscr();

menuborder();

print\_centre(" HIGHSCORE ",6);

print\_centre("1. EASY",8);

print\_centre("2. DIFFICULT",11);

print\_centre("3. HARD",14);

print\_centre("Enter your Choice : ",18);

cin>>chh;

clrscr();

scoreborder();

bfprint(chh);

getch();

break;

case 4:exit(0);

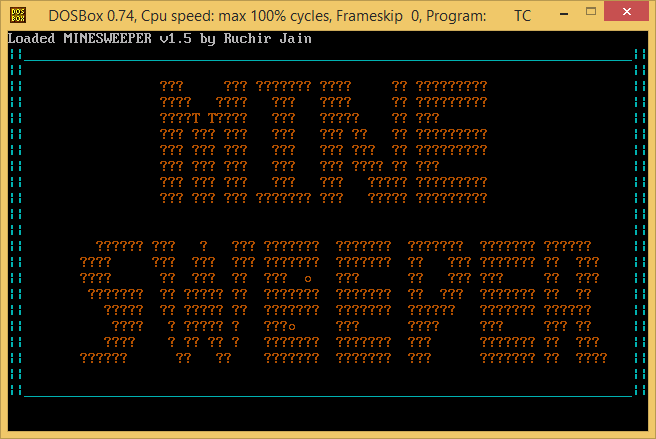
}

}while(ch!=4);

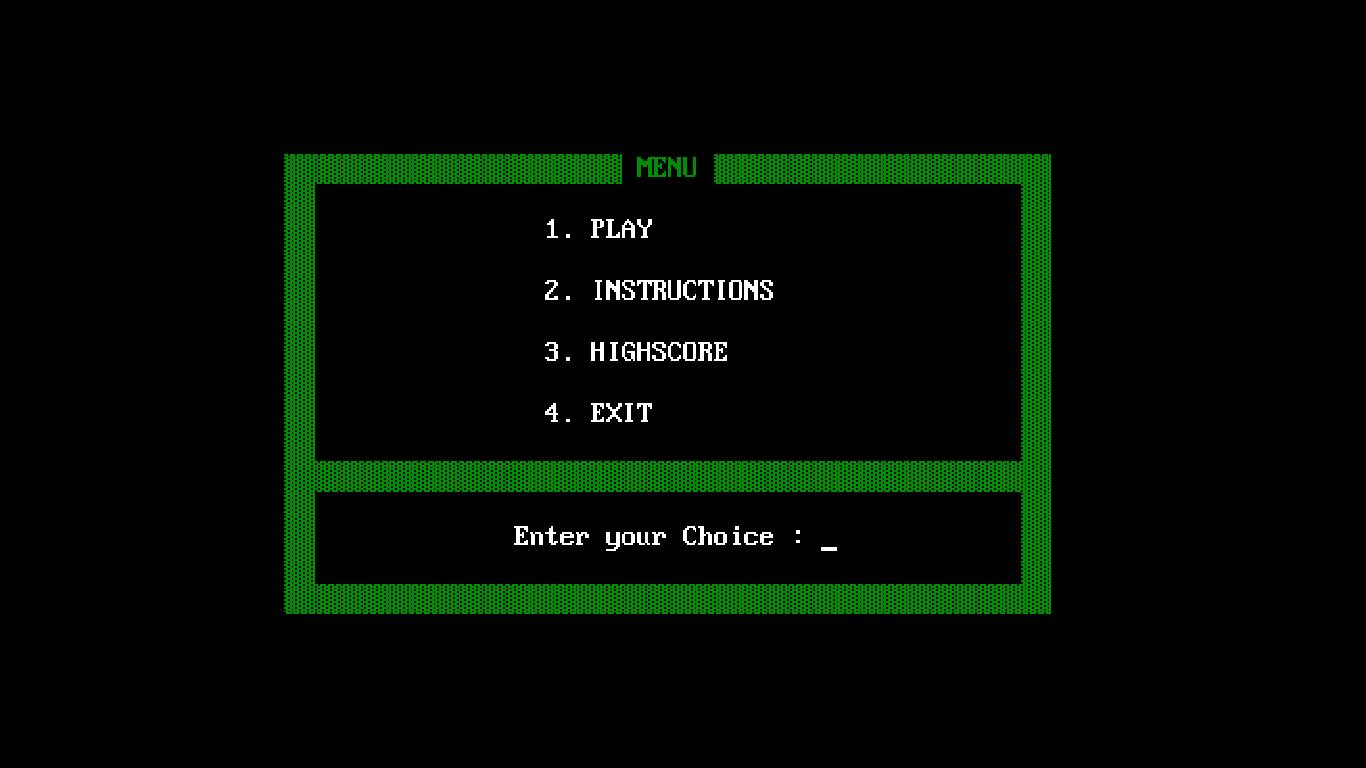
}

***Output Screens***

***Welcome Screen***

******

***Main Screen***

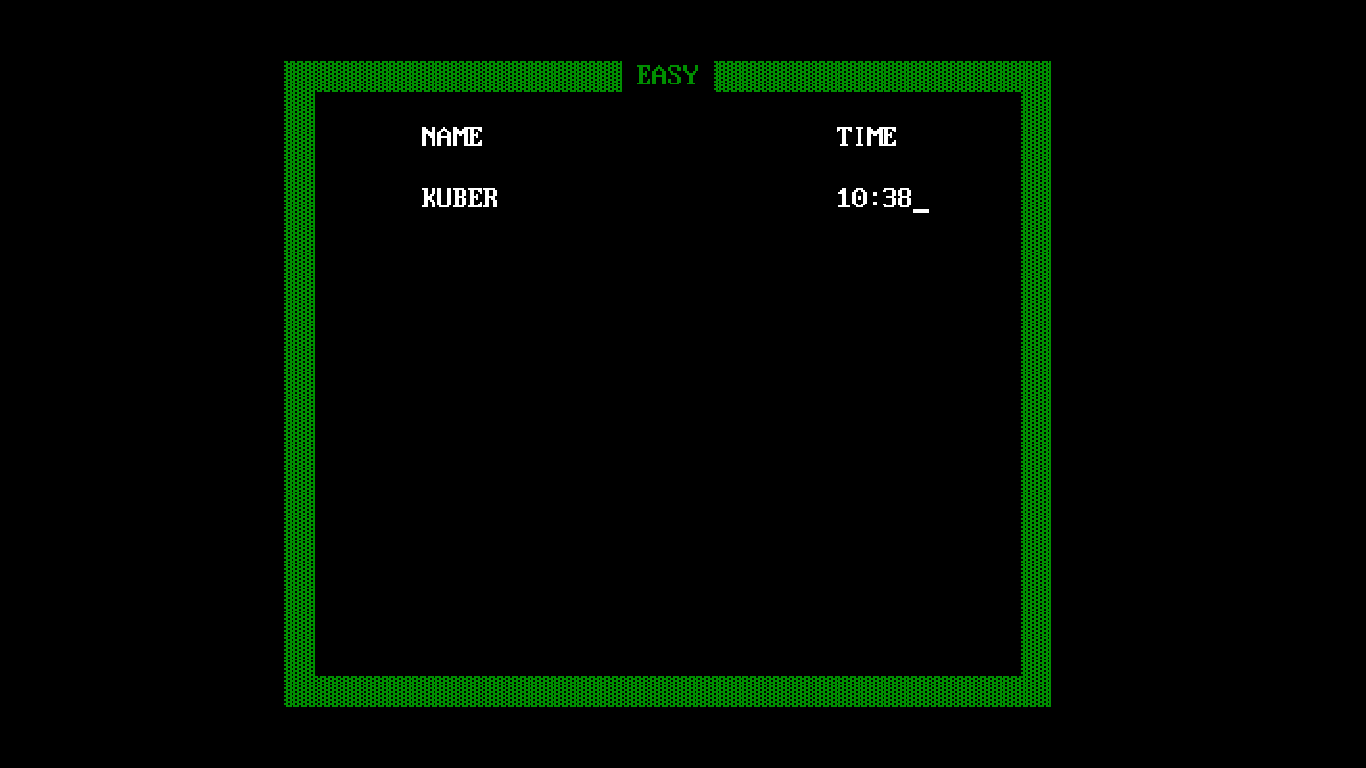


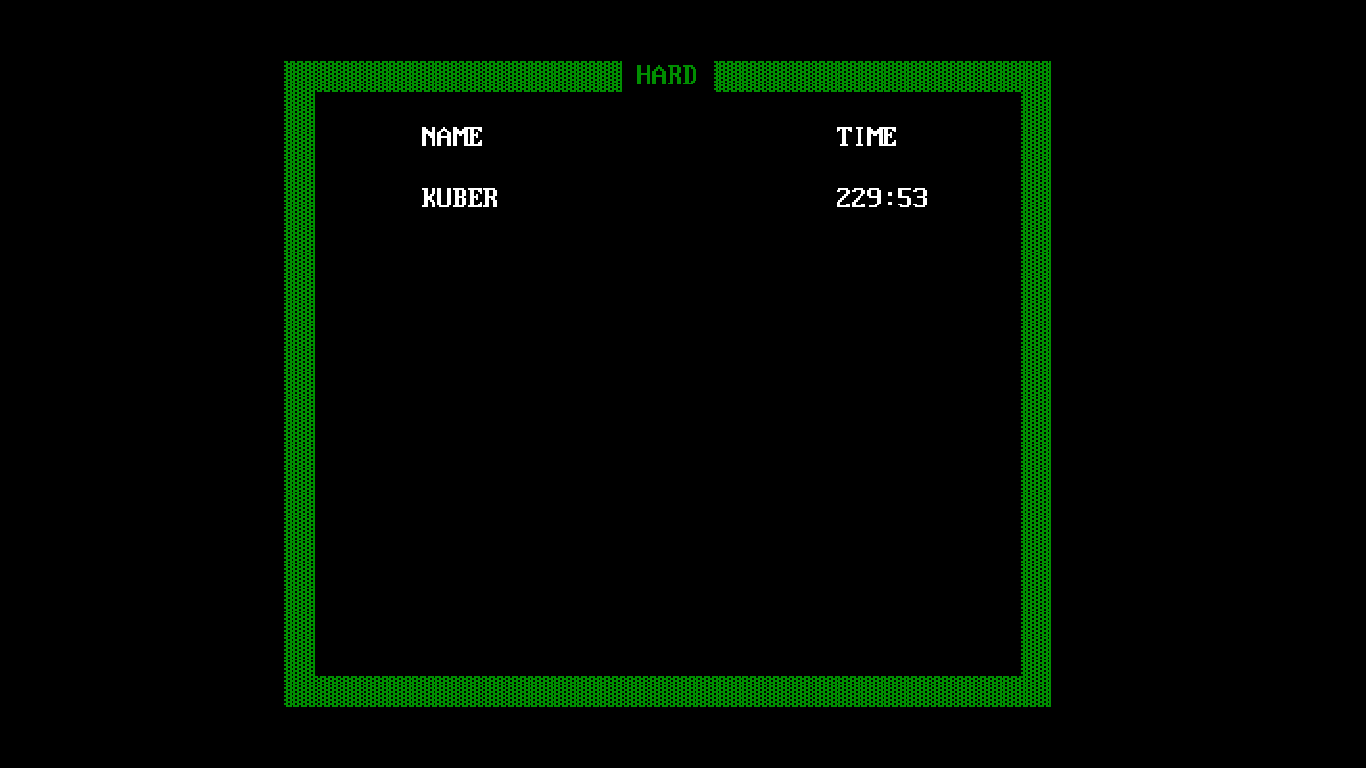
***Instructions***



***Viewing the highscore***

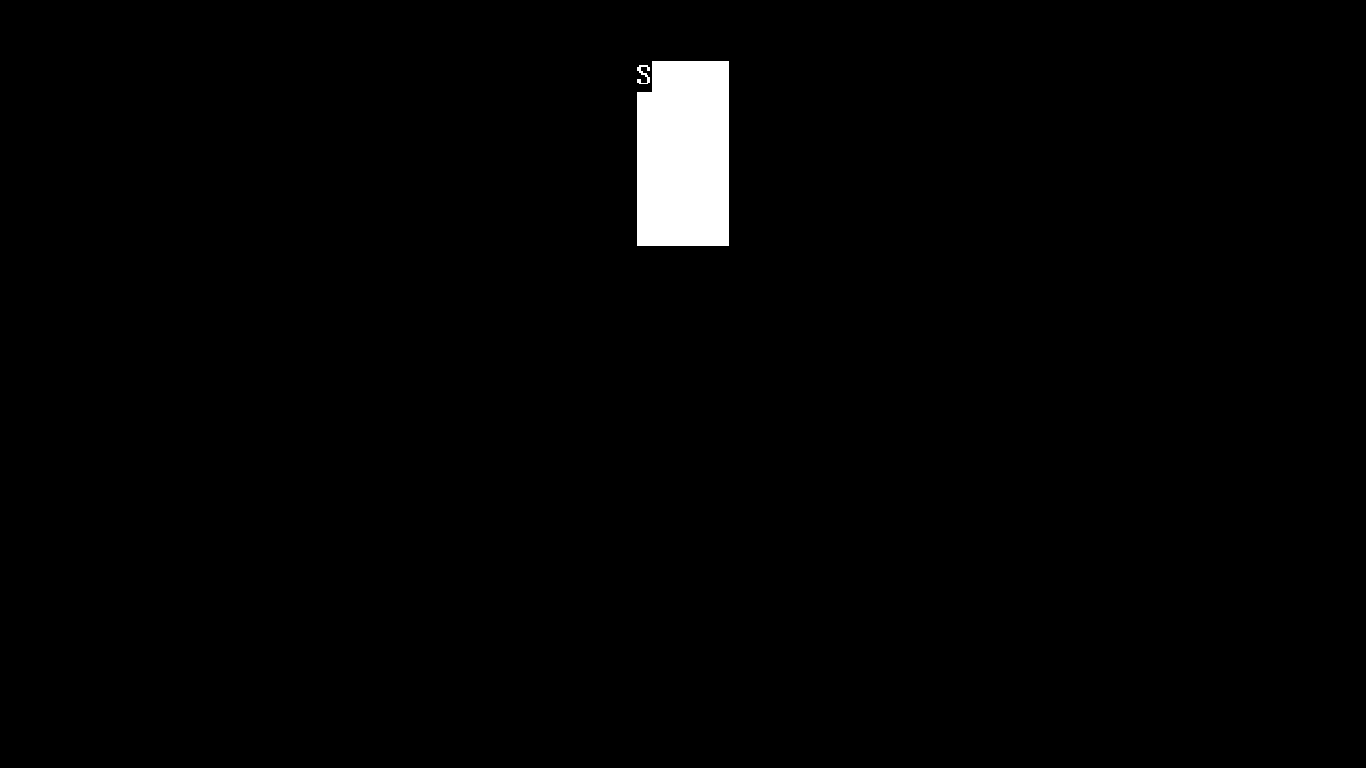


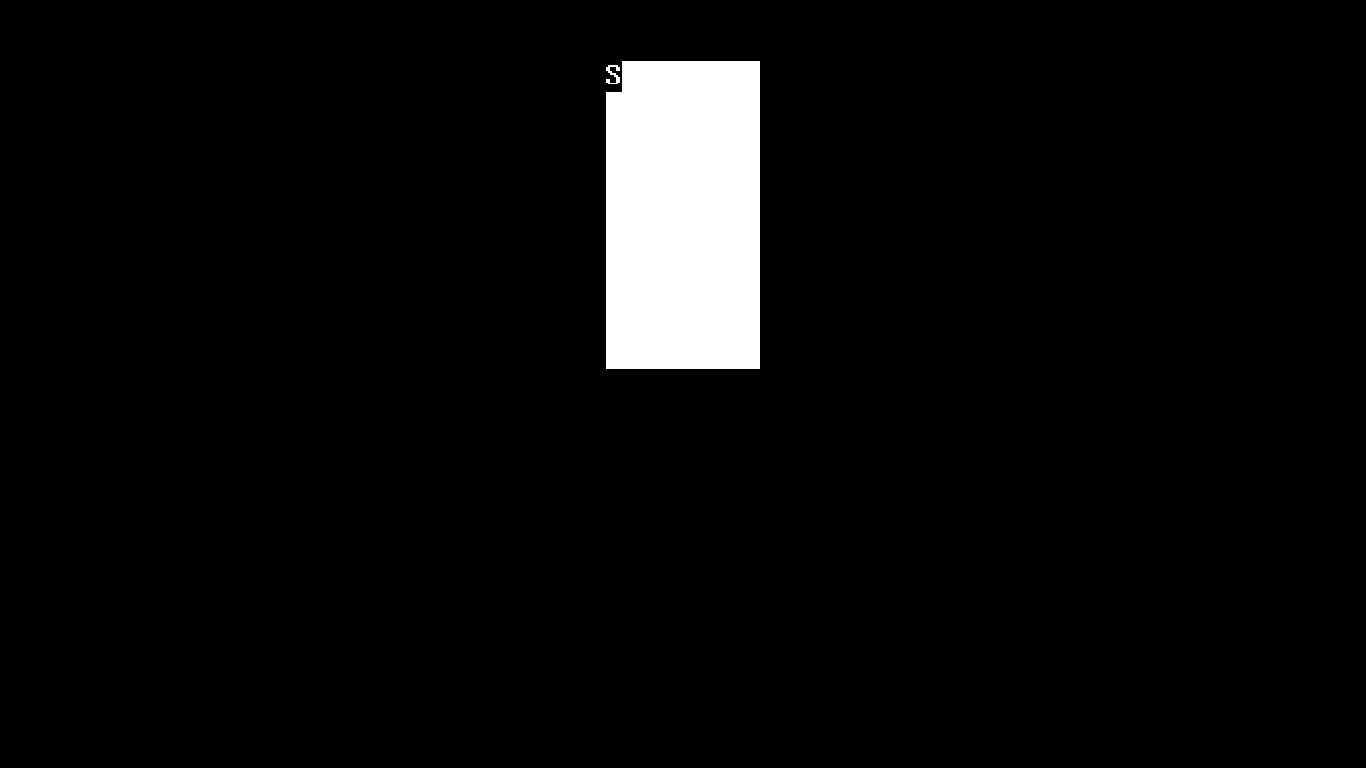


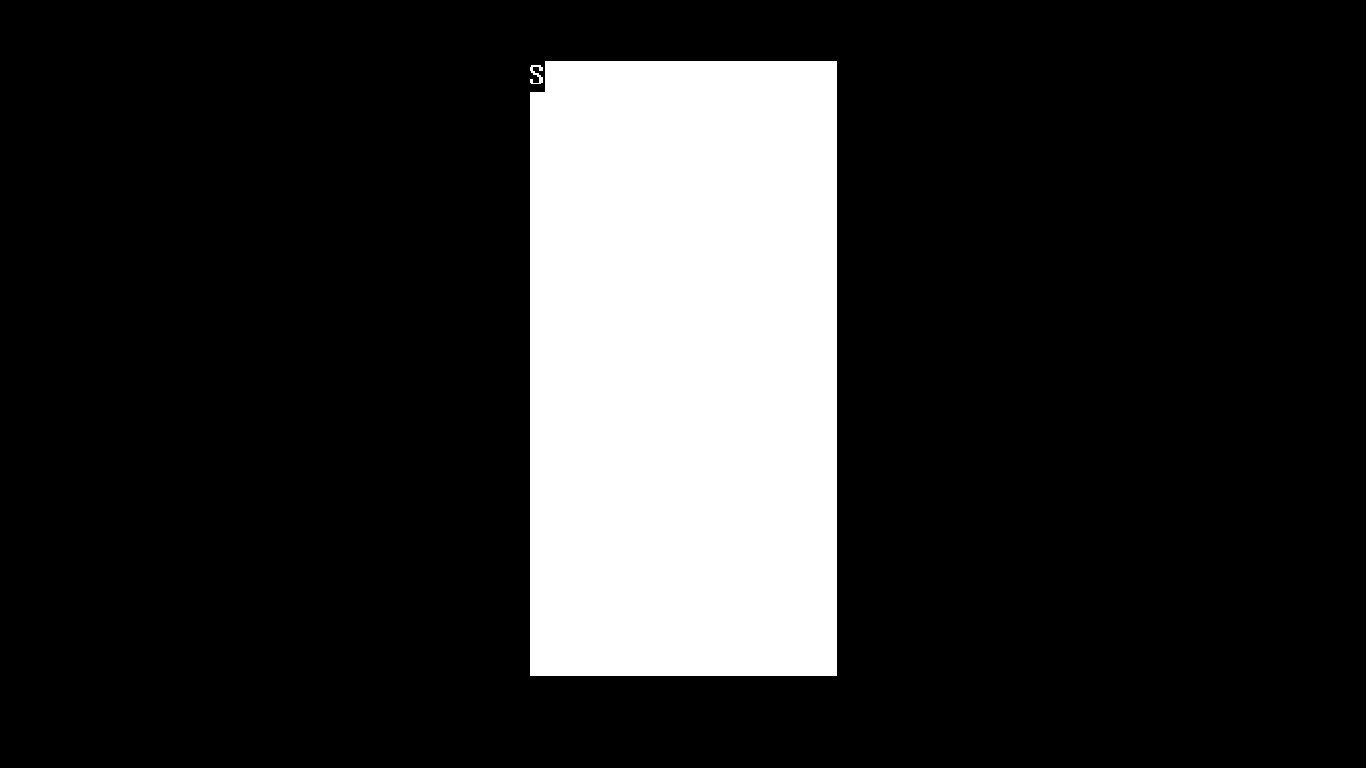
******

***The 3 difficulty levels***

******

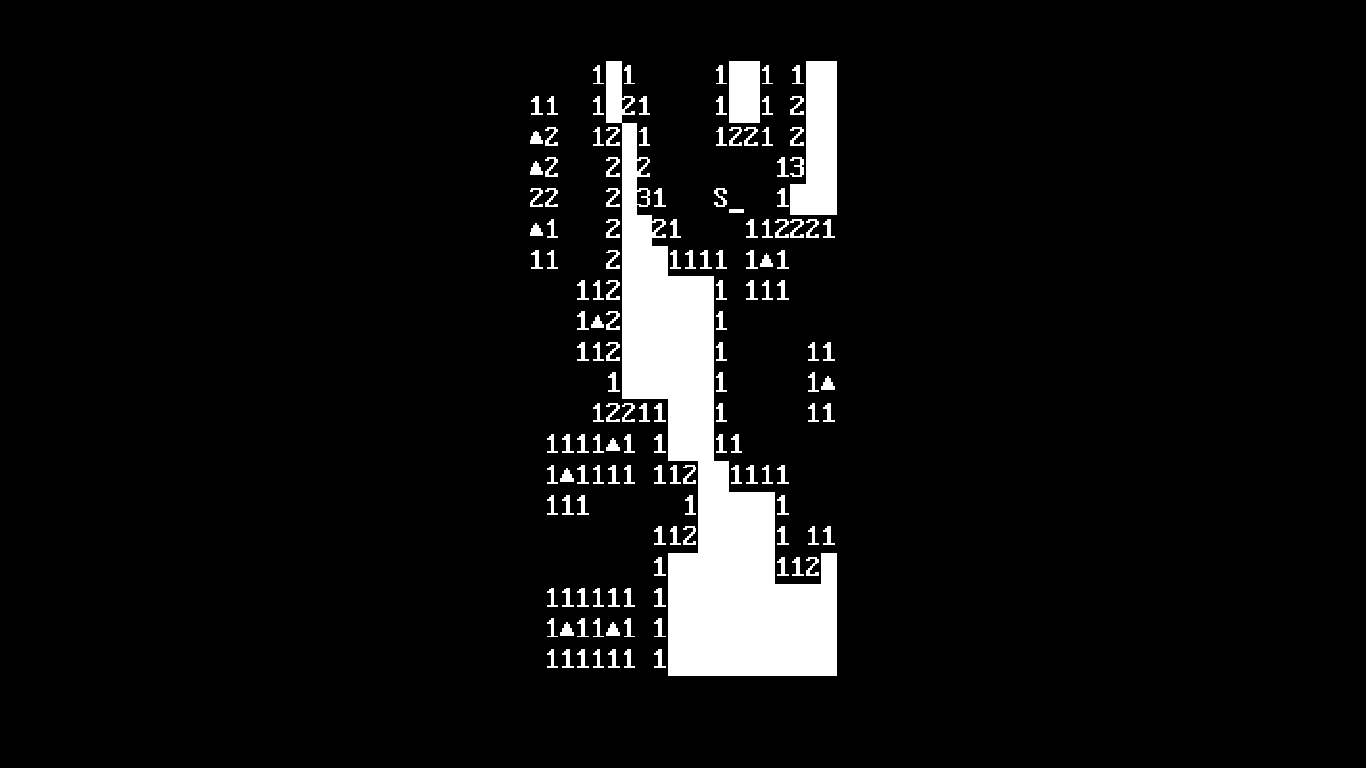
******

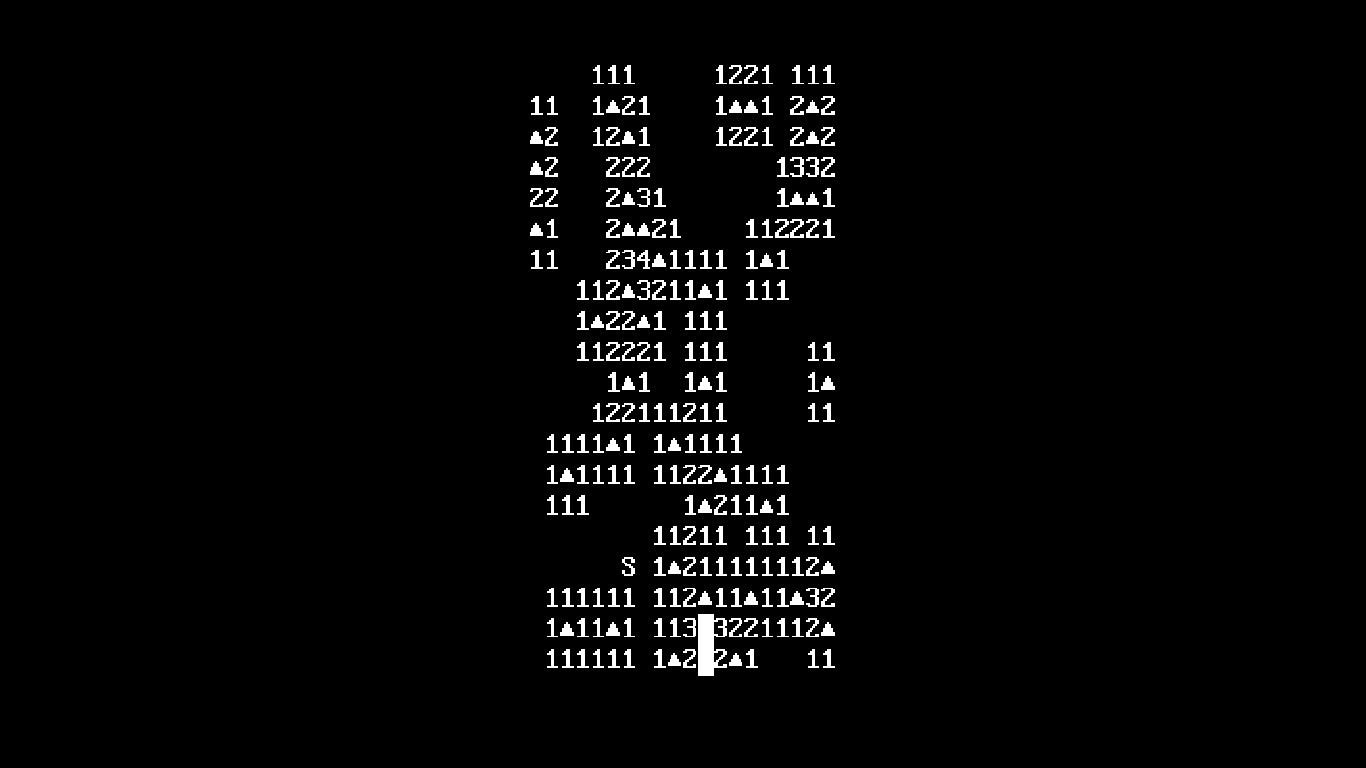
******

******

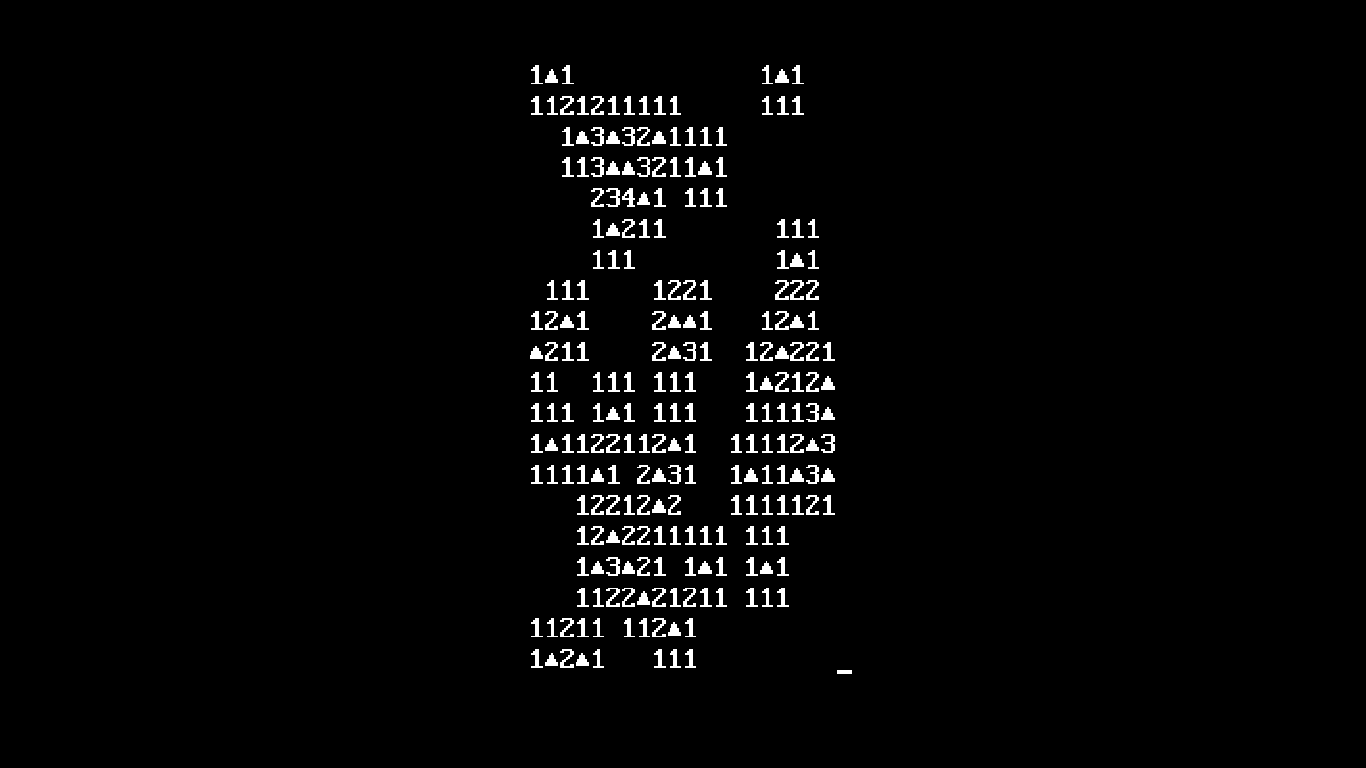
***Playing the game***

******

******

******

**Pressing ‘P’ for solution**

******

***Acknowledgement***

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion of my project work. Whatever I have done is only due to such guidance and assistance and I would not forget to thank them.

I respect and thank Mrs. Ritu Nagpal, for giving me an opportunity to do this project and providing me all support and guidance which made me complete the project on time. I am extremely grateful to her for providing such a nice support and guidance.

I owe my profound gratitude to Mr. Mervin Fernandes, who took keen interest on my project work and guided me all along, till the completion of my project work by providing all the necessary information.