2018-2019

RUCHIR JAIN

MINESWEEPER

St Coloumbas' School | C++ project

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

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 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
```

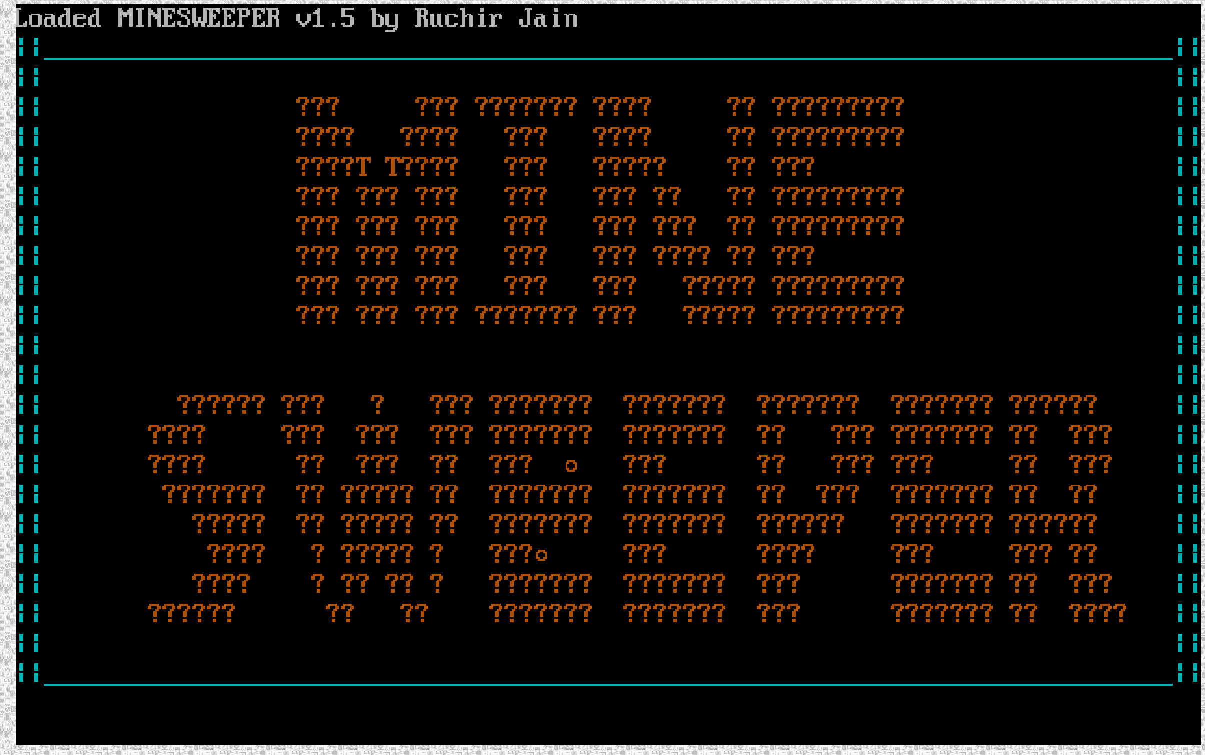
```
  cprintf("ｲｲ");
 for(i=1;i<20;i++)//Left border
 {
  gotoxy(15,i+3);
  cprintf("ｲｲ");
 }
 gotoxy(15,23);
 for(i=0;i<25;i++)//Lower border
  cprintf("ｲｲ");
 for(i=1;i<20;i++)
 {
  gotoxy(63,i+3);
  cprintf("ｲｲ");
 }
 textcolor(WHITE);
 gotoxy(24,5);
 cprintf("NAME");
 gotoxy(51,5);
 cprintf("TIME");
}
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);
}
```

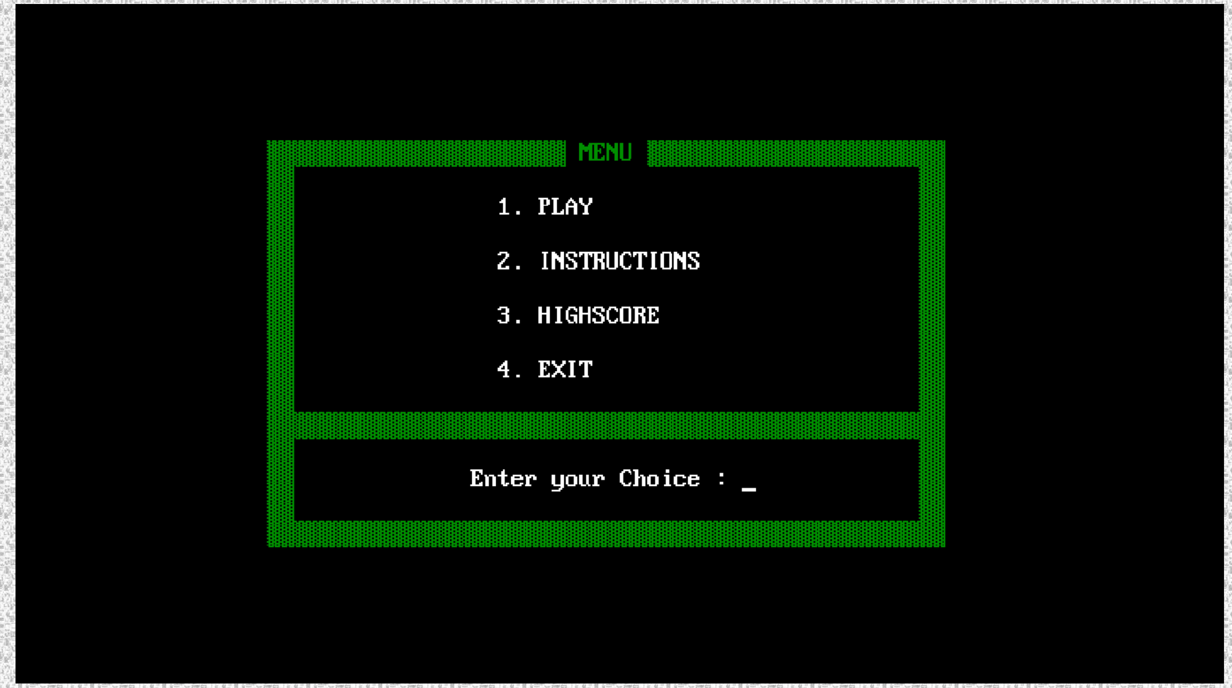
Output Screens

Welcome Screen



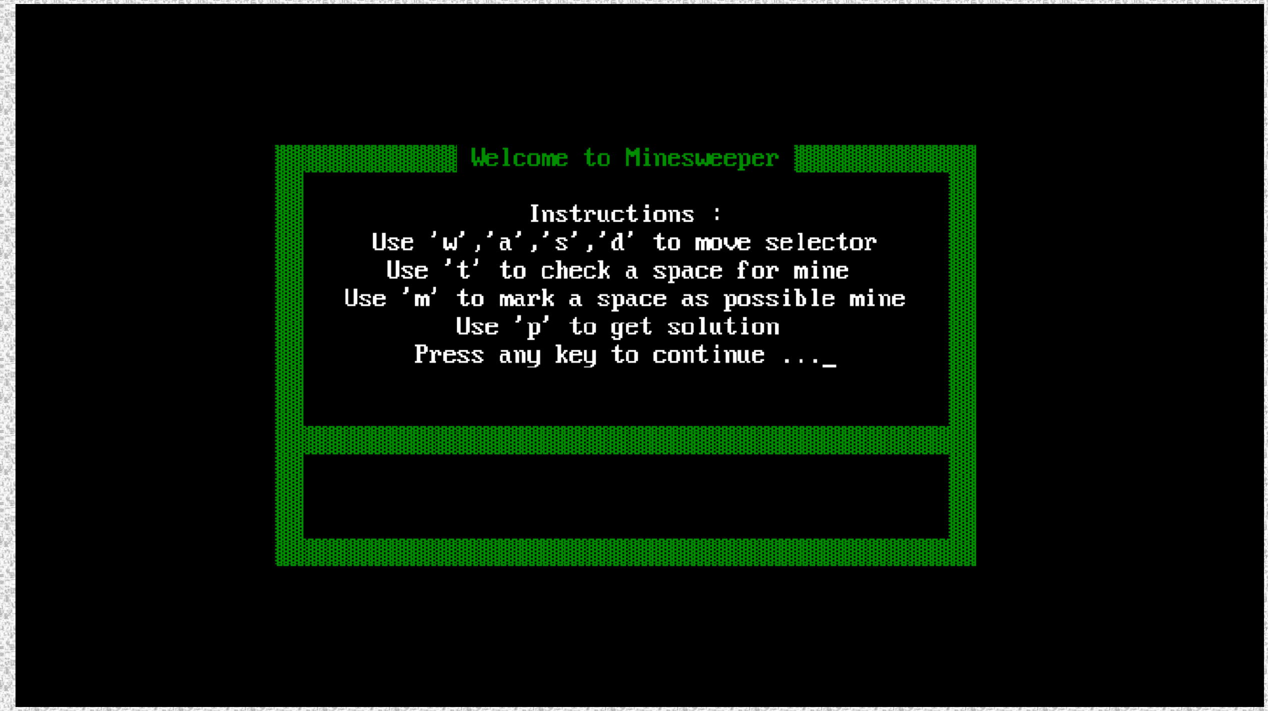
```
Loaded MINESWEEPER v1.5 by Ruchir Jain

                  ???     ??? ??????? ????     ?? ?????????
                  ????   ????   ???   ????     ?? ?????????
                  ????T T????   ???   ?????    ?? ???
                  ??? ??? ???   ???   ??? ??   ?? ?????????
                  ??? ??? ???   ???   ??? ???  ?? ?????????
                  ??? ??? ???   ???   ??? ???? ?? ???
                  ??? ??? ???   ???   ???   ????? ?????????
                  ??? ??? ??? ??????? ???   ????? ?????????


          ?????? ???   ?   ??? ???????  ???????  ???????  ??????? ??????
        ????     ???  ???  ??? ???????  ???????  ??   ??? ??????? ??  ???
        ????      ??  ???  ??  ???  o   ???      ??   ??? ???     ??  ???
         ???????  ?? ????? ??  ???????  ???????  ??  ???  ??????? ??  ??
           ?????  ?? ????? ??  ???????  ???????  ??????   ??????? ??????
            ????   ? ????? ?   ???o     ???      ????     ???     ??? ??
           ????    ? ?? ?? ?   ???????  ???????  ???      ??????? ??  ???
        ??????      ??   ??    ???????  ???????  ???      ??????? ??  ????
```

Main Screen



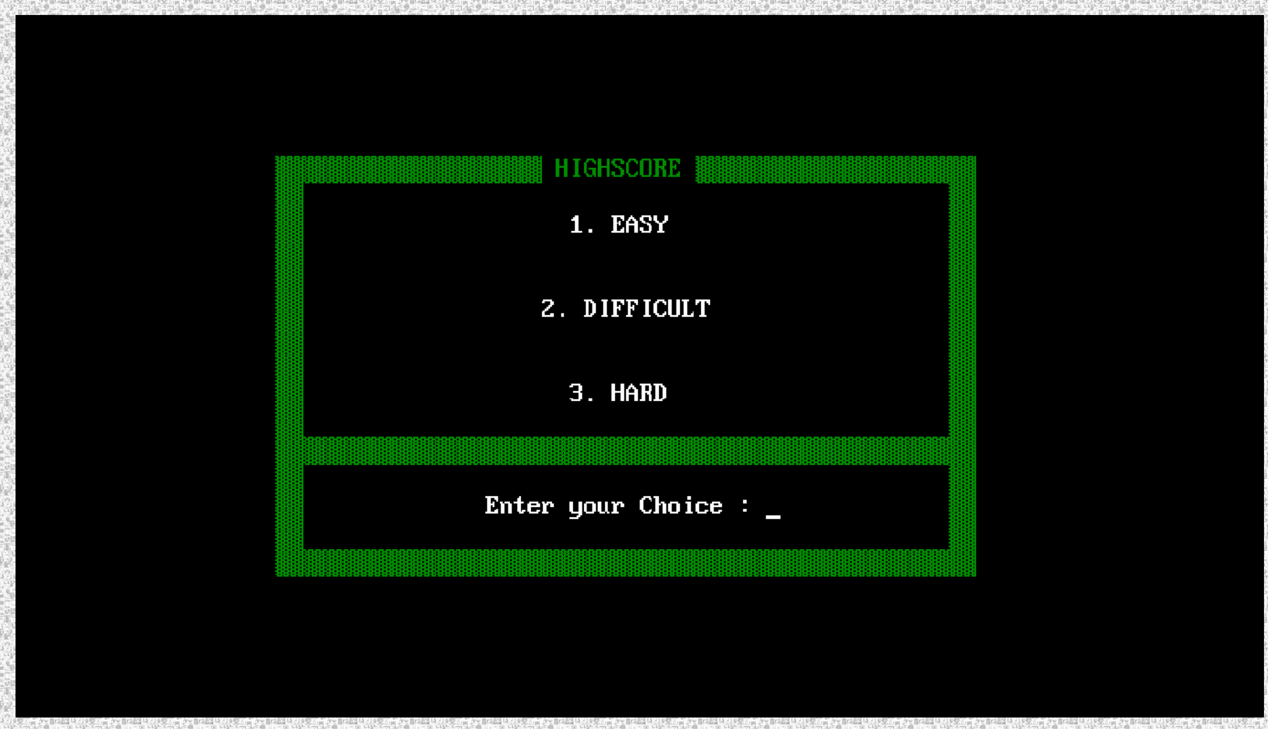
```
                 MENU

             1. PLAY

             2. INSTRUCTIONS

             3. HIGHSCORE

             4. EXIT


         Enter your Choice : _
```

Instructions

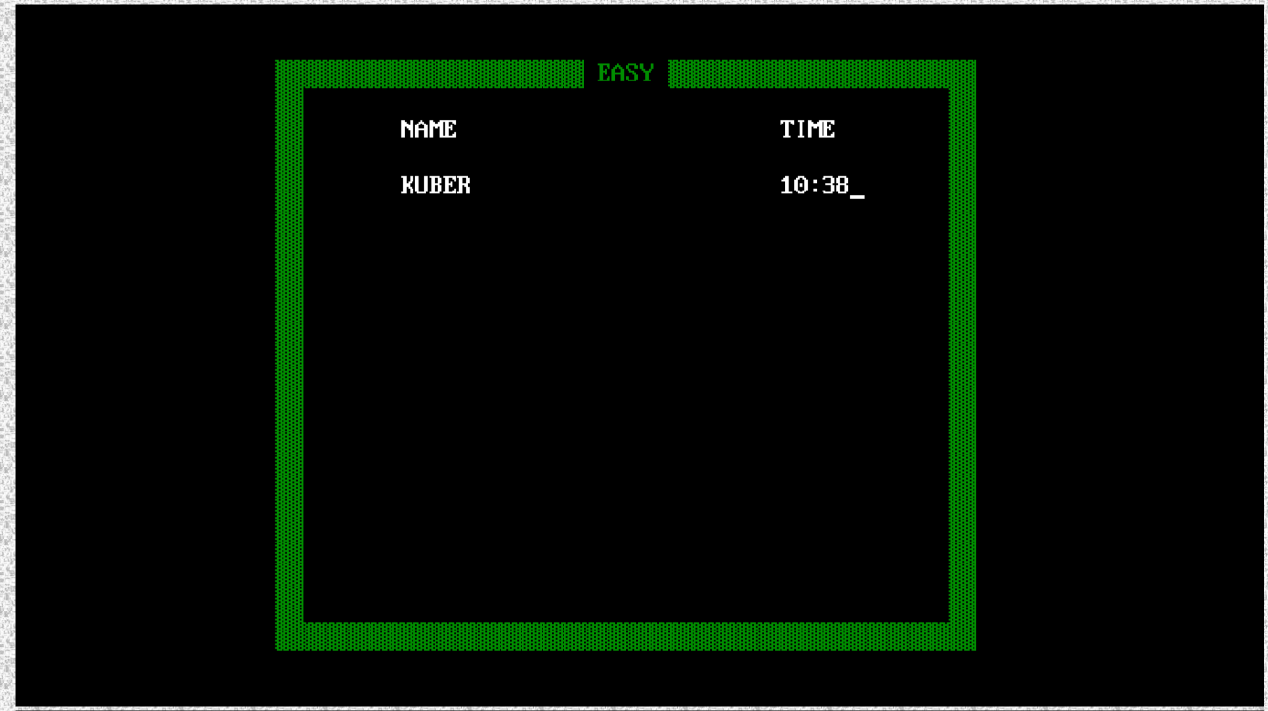


```
Welcome to Minesweeper

Instructions :
Use 'w','a','s','d' to move selector
Use 't' to check a space for mine
Use 'm' to mark a space as possible mine
Use 'p' to get solution
Press any key to continue ..._
```

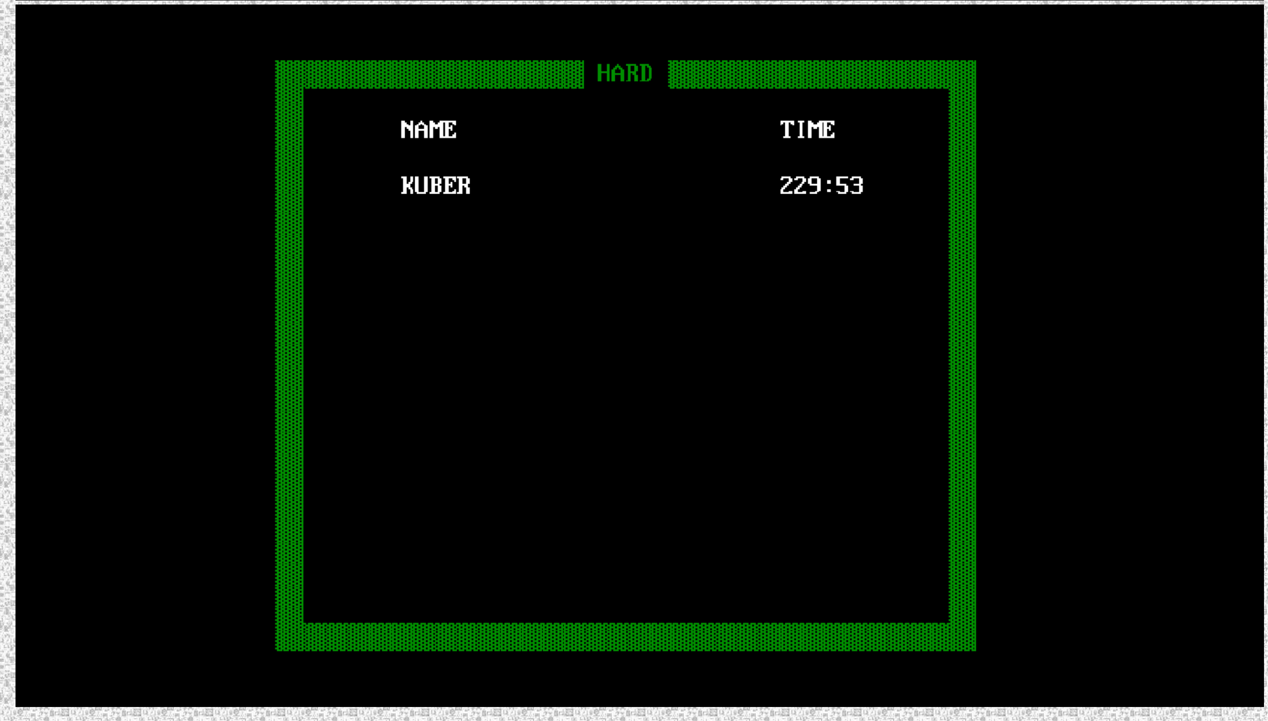
Viewing the highscore



```
HIGHSCORE

1. EASY

2. DIFFICULT

3. HARD

Enter your Choice : _
```

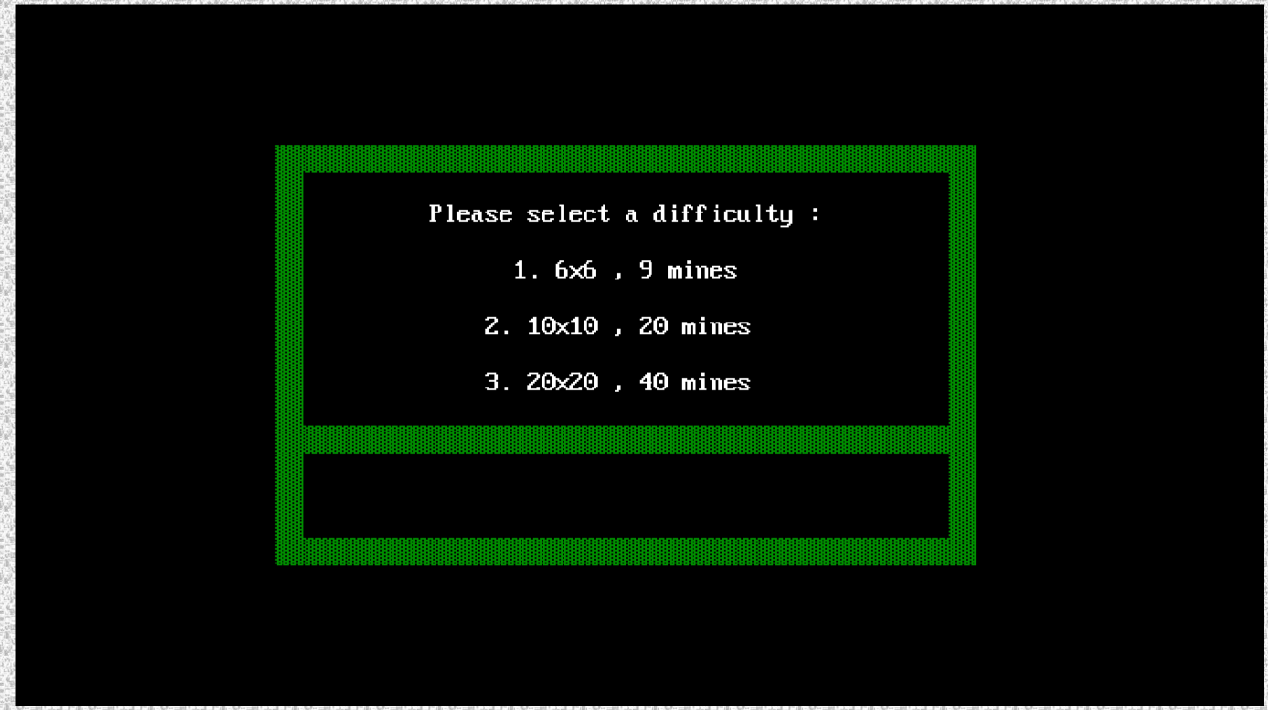


```
EASY

NAME                         TIME

KUBER                        10:38_
```

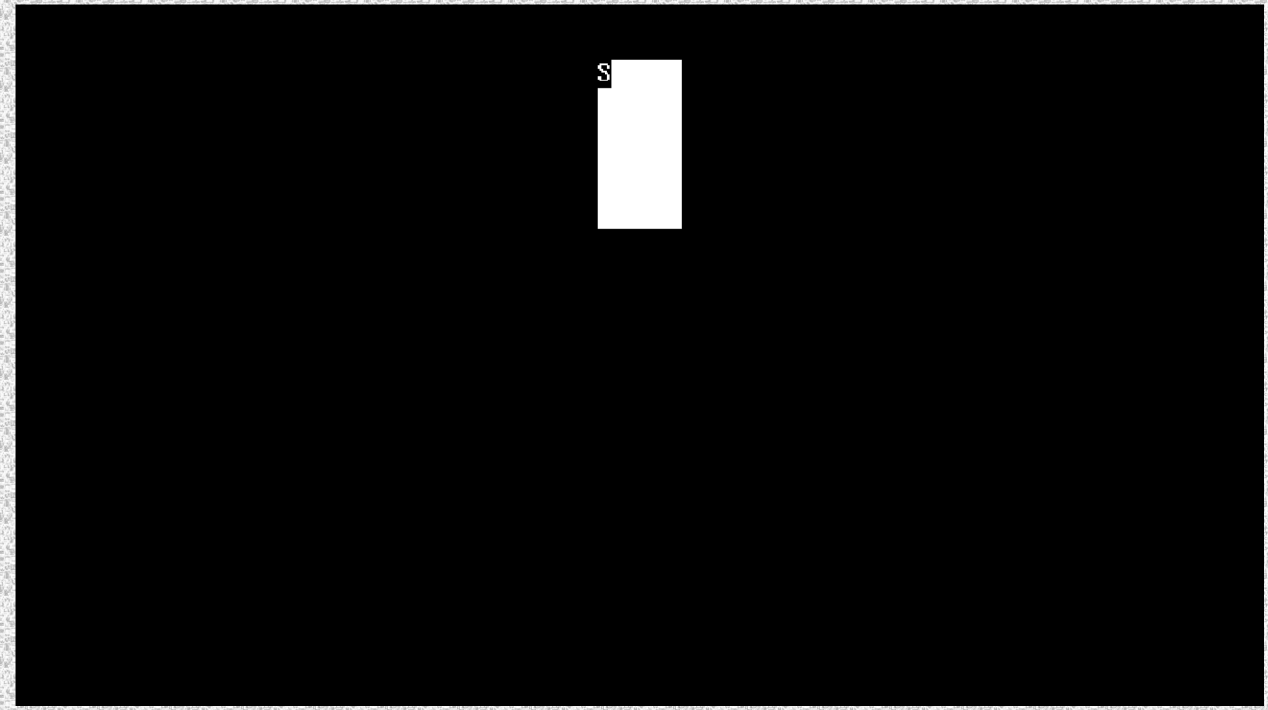


```
HARD

NAME                         TIME

KUBER                        229:53
```

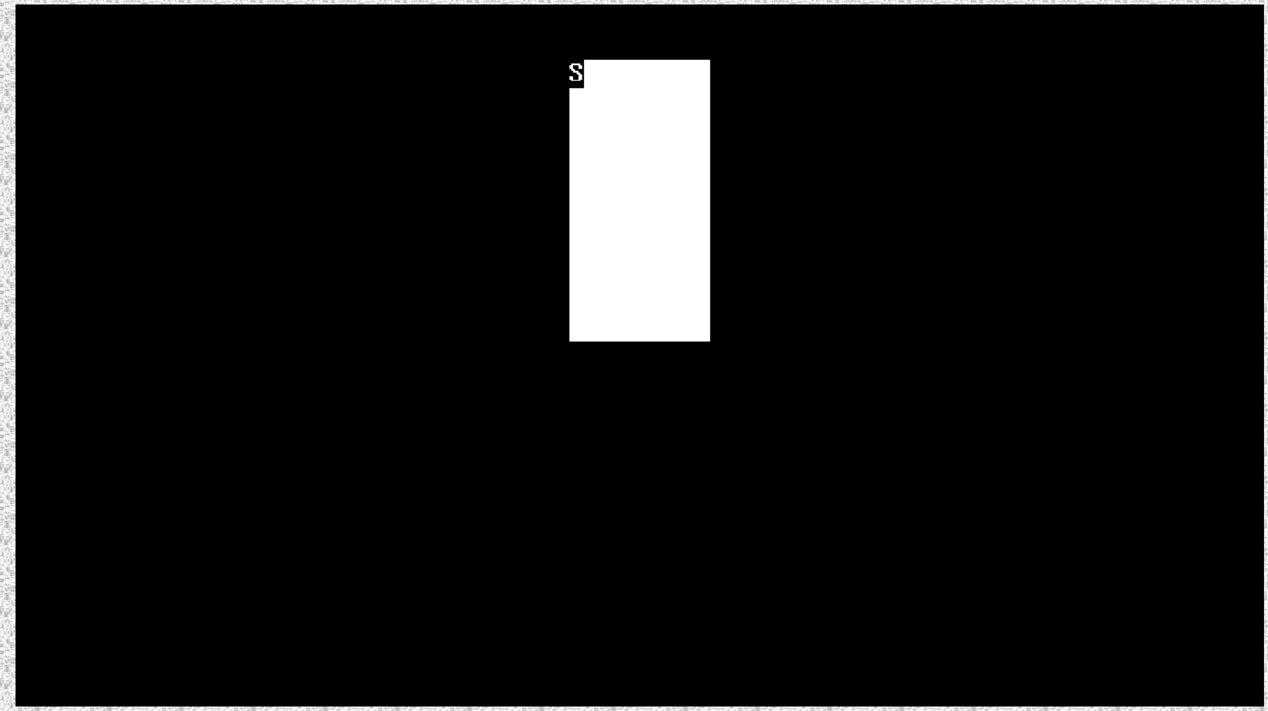
The 3 difficulty levels

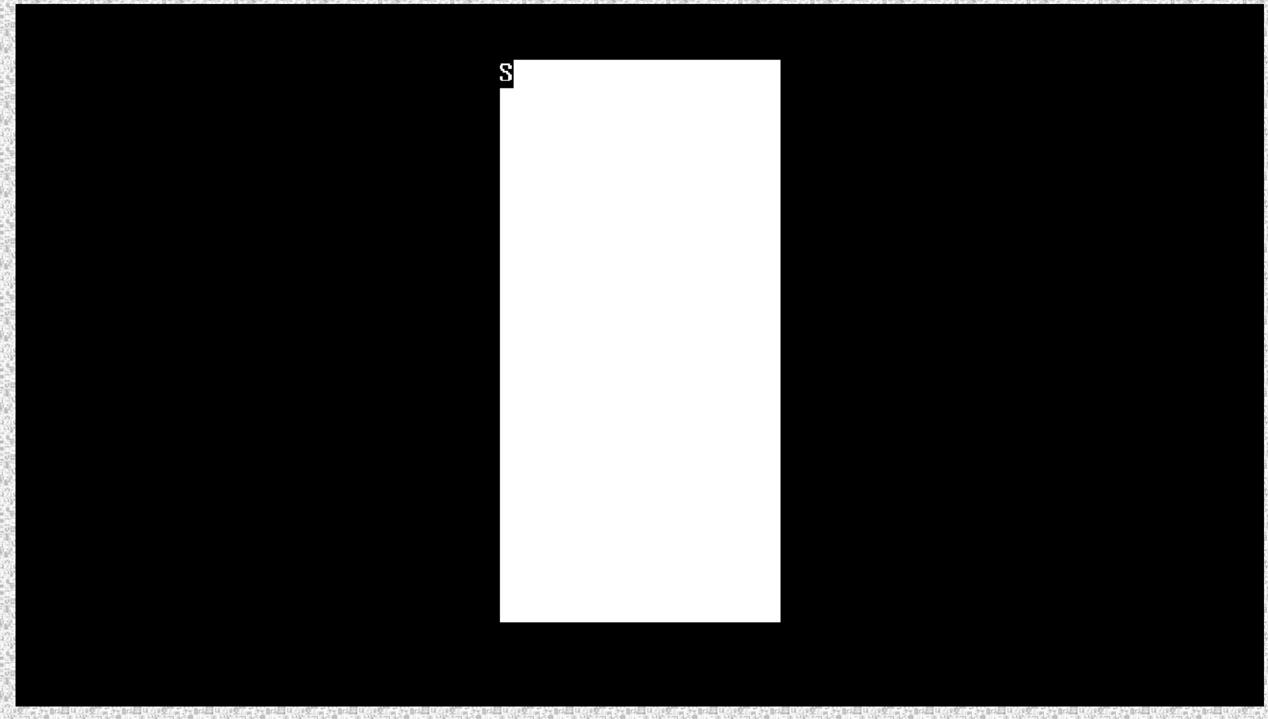


```
Please select a difficulty :

1. 6x6 , 9 mines

2. 10x10 , 20 mines

3. 20x20 , 40 mines
```

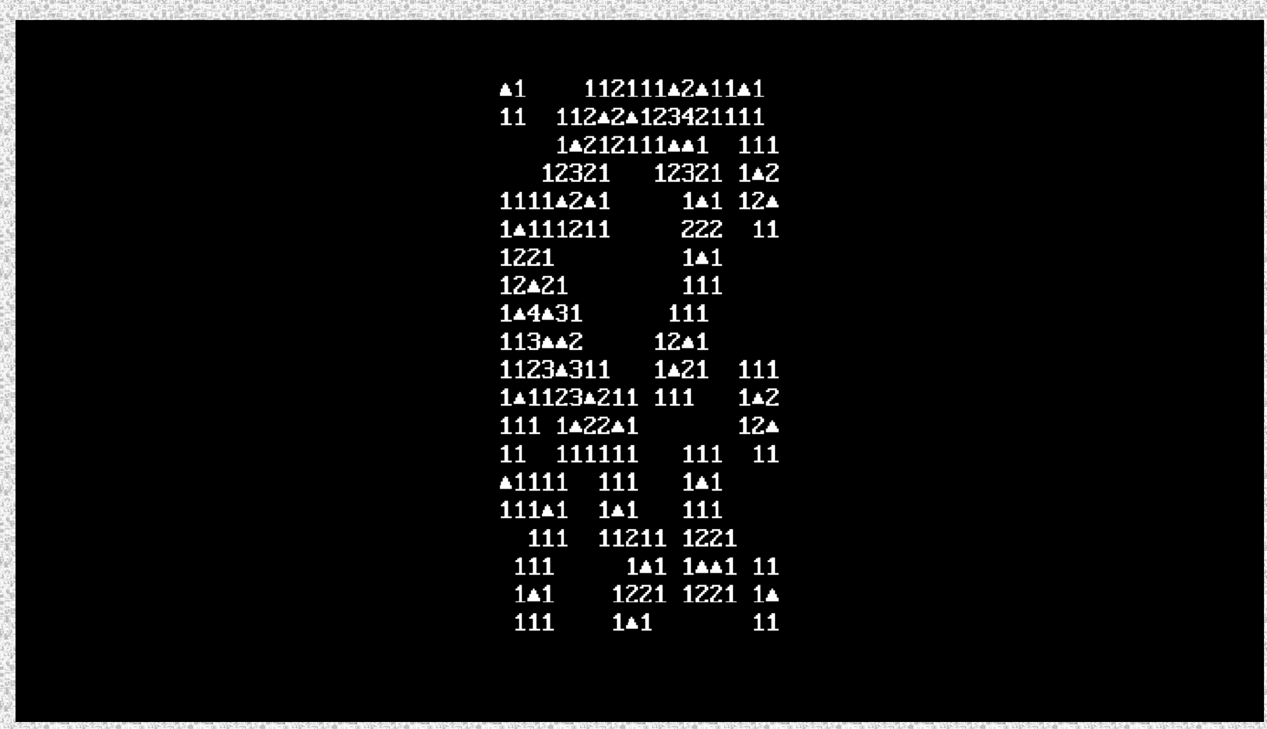


```
S
```

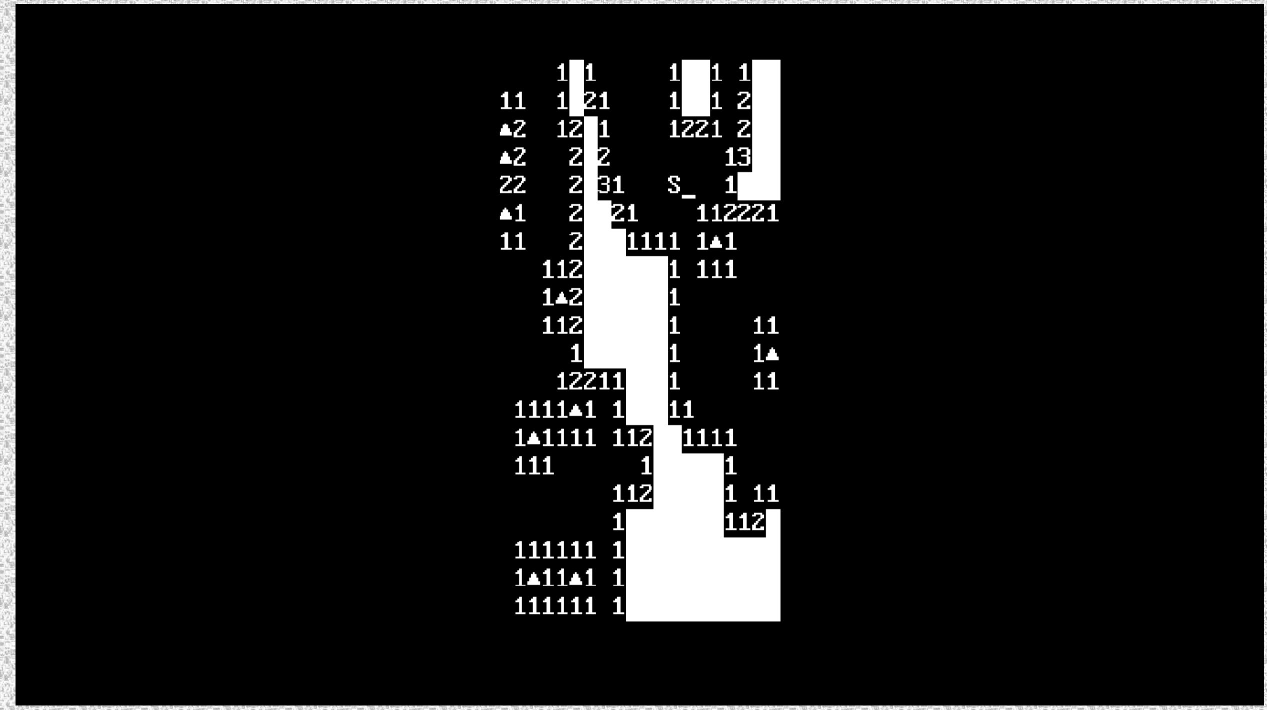

S


S

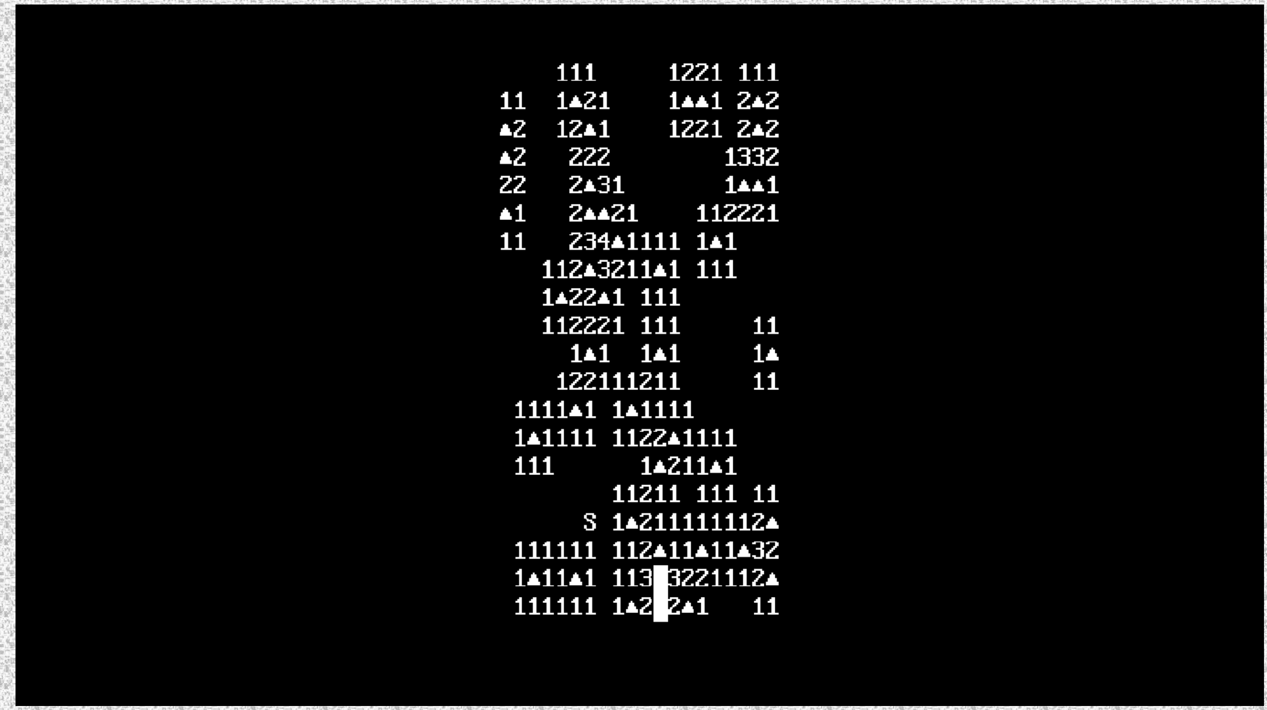
Playing the game



```
▲1    112111▲2▲11▲1
11  112▲2▲123421111
    1▲212111▲▲1  111
   12321   12321 1▲2
1111▲2▲1     1▲1 12▲
1▲111211     222  11
1221         1▲1
12▲21        111
1▲4▲31      111
113▲▲2     12▲1
1123▲311   1▲21  111
1▲1123▲211 111   1▲2
111 1▲22▲1       12▲
11  111111   111  11
▲1111  111   1▲1
111▲1  1▲1   111
  111  11211 1221
 111      1▲1 1▲▲1 11
 1▲1    1221 1221 1▲
 111    1▲1         11
```

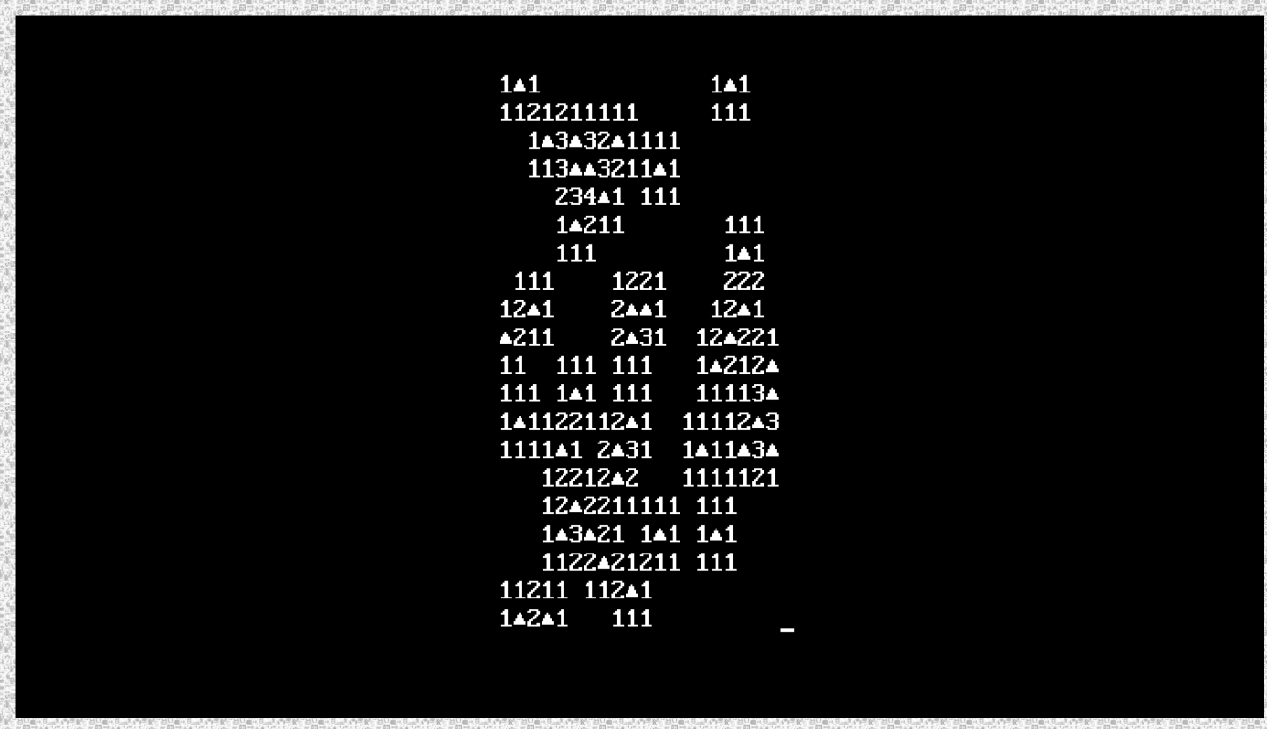


```
    1 1     1  1 1
11  1 21    1  1 2
▲2  12 1    1221 2
▲2   2 2        13
22   2 31   S_  1
▲1   2  21    112221
11   2   1111 1▲1
   112      1 111
   1▲2      1
   112      1     11
     1      1     1▲
    12211   1     11
 1111▲1 1   11
 1▲1111 112  1111
 111      1     1
        112     1 11
        1       112
 111111 1
 1▲11▲1 1
 111111 1
```



```
    111     1221 111
11  1▲21    1▲▲1 2▲2
▲2  12▲1    1221 2▲2
▲2   222        1332
22   2▲31       1▲▲1
▲1   2▲▲21    112221
11   234▲1111 1▲1
   112▲3211▲1 111
   1▲22▲1 111
   112221 111     11
     1▲1  1▲1     1▲
    122111211     11
 1111▲1 1▲1111
 1▲1111 1122▲1111
 111      1▲211▲1
        11211 111 11
      S 1▲211111112▲
 111111 112▲11▲11▲32
 1▲11▲1 113█3221112▲
 111111 1▲2█2▲1   11
```

Pressing ‘P’ for solution



```
1▲1            1▲1
1121211111     111
  1▲3▲32▲1111
  113▲▲3211▲1
    234▲1 111
    1▲211       111
    111         1▲1
 111    1221    222
12▲1    2▲▲1   12▲1
▲211    2▲31  12▲221
11  111 111   1▲212▲
111 1▲1 111   11113▲
1▲1122112▲1  11112▲3
1111▲1 2▲31  1▲11▲3▲
   12212▲2   1111121
   12▲2211111 111
   1▲3▲21 1▲1 1▲1
   1122▲21211 111
11211 112▲1
1▲2▲1   111         _
```

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.