" ****_MINESWEEPER GAME_***** "

A MINI- PROJECT REPORT ON

Submitted in partial fulfillment of the requirements

For the degree of

Bachelor of Engineering

In

Information Technology

by

Name of the Student-1: SAISH SHINDE	**Roll No.-18IT1035**
Name of the Student-2: UNNATI SHIROLE	**Roll No.-18IT1045**
Name of the Student-3: RANJEET PATIL	**Roll No.-18IT2036**
Name of the Student-4: AVINASH PARANDEKAR	**Roll No-18IT1084**

Supervisor

Nilima Dongre

Department of Information Technology

Dr. D. Y. Patil Group's

Ramrao Adik Institute of Technology

Dr. D. Y. Patil Vidyanagar, Sector 7, Nerul, Navi Mumbai 400706.
(Affiliated to University of Mumbai)

Ramrao Adik Institute of Technology

(2020)

D Y PATIL
RAMRAO ADIK
INSTITUTE OF
TECHNOLOGY
NAVI MUMBAI

Ramrao Adik Institute of Technology

(Affiliated to the University of Mumbai)
Dr. D. Y. Patil Vidyanagar,Sector 7, Nerul, Navi Mumbai 400706.

CERTIFICATE

This is to certify that,Mini Project entitled

"MINESWEEPER GAME USING BASH SCRIPTING"

is a bonafide work done by

Student Names
1. SAISH SHINDE
2. UNNATI SHIROLE
3. RANJEET PATIL
4. AVINASH PARNADEKAR

and is submitted in the partial fulfillment of the requirement for the degree of
Bachelor of Engineering
in
Information Technology
to the
University of Mumbai

Supervisor
Prof.Nilima M.Dongre

Ramrao Adik Institute of Technology

Project Guide   Head of the department

Nilima Dongre   Dr. Ashish Jadhav

Certificate of Approval by Examiners

This Mini Project report entitled " Minesweeper Game " is a bonafide work done by Student Names under the supervision of Prof.Nilima Dongre approved for the award of Bacheor Degree in Information Technology, University of Mumbai.

Examiners :

1.................................

2.................................

Supervisors :

1.................................

2.................................

Principal :

....

Date :

Place :

Ramrao Adik Institute of Technology

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name and Roll No. of Students   **Signature**

1. SAISH SHINDE(18IT1035)
2. UNNATI SHIROLE(18IT1045)
3. RANJEET PATIL(18IT2036)
4 .AVINASH PARANDEKAR(18IT1084)

Date:

Place:

Ramrao Adik Institute of Technology

ACKNOWLEDGEMENT

The project "Minesweeper Game" is creative work of many minds. A proper synchronization between individual is must for any project to be completed successfully. One cannot imagine the power of the force that guides us all and neither can we succeed without acknowledging it.

We take this opportunity to express my profound gratitude and deep regards to our Guide **Nilima Dongre**, Department of the Information Technology Engineering for her or her exemplary guidance, monitoring and constant encouragement throughout the completion of this mini project.

We would like to express our gratitude to **Dr. Ashish Jadhav,** Head of the department, Information Technology Engineering for encouraging and inspiring us to carry out the project in the department lab. We take this privilege to express my sincere thanks are thankful to **Dr. Mukesh D. Patil, Principal RAIT,** for his constant support and motivation.

We also would like to thank all the staff members Department of the Information Technology Engineering for providing us with the required facilities and support towards the completion of the project.

Last but not the least we are thankful to our parents and friends for their constant Inspiration, encouragement and well wishes by which we have made a challenging project.

STUDENT- UNNATI . G. SHIROLE (18IT1045)   Signature

PREFACE

We take great opportunity to present this Mini Project report on "**MINESWEEPER GAME"** and put before readers some useful information regarding our project.

We have made sincere attempts and taken every care to present this matter in precise and compact form, the language being as simple as possible. We are sure that the information contained in this volume certainly prove useful for better insight in the scope and dimension of this project in it true perspective.

The task of the completion of the project though being difficult was made quite simple, interesting and successful due to deep involvement and complete dedication of our group members.

TABLE OF CONTENTS

Ramrao Adik Institute of Technology

TABLE OF CONTENTS

ABSTRACT

Our mini-project that we present here today is 'Minesweeper Game'. But the question that comes to our mind is what is exactly is a 'Minesweeper game?
- WHAT IS MINESWEEPER GAME?

Minesweeper is a single-player puzzle video game. The objective of the game is to clear a rectangular board containing hidden "mines" or bombs without detonating any of them, with help from clues about the number of neighbouring mines in each field. The game originates from the 1960s, and has been written for many computing platforms in use today. It has many variations and offshoots.

Some versions of Minesweeper will set up the board by never placing a mine on the first square revealed. Minesweeper for versions of Windows protects the first square revealed; from Windows 7 onward, players may elect to replay a board, in which the game is played by revealing squares of the grid by clicking or otherwise indicating each square. If a square containing a mine is revealed, the player loses the game. If no mine is revealed, a digit is instead displayed in the square, indicating how many adjacent squares contain mines; if no mines are adjacent, the square becomes blank, and all adjacent squares will be recursively revealed. The player uses this information to deduce the contents of other squares, and may either safely reveal each square or mark the square as containing a mine.

Ramrao Adik Institute of Technology

CHAPTER -1

INTRODUCTION

INTRODUCTION

1.1 INTRODUCTION TO SCRIPTING LANGUAGES

Usually shells are interactive that mean, they accept command as input from users and execute them. However some time we want to execute a bunch of commands routinely, so we have type in all commands each time in terminal. Shell scripts are similar to the batch file in MS-DOS. Each shell script is saved

with .sh file extension eg . myscript.sh A shell script have syntax just like any other programming language. If you have any prior experience with any programming language like Python, C/C++ etc. it would be very easy to get started with it.

1.2 WHY PARTICULAR SCRIPTING LANGUAGE

- There are many reasons to write shell scripts –
- To avoid repetitive work and automation
- System admins use shell scripting for routine backups
- System monitoring
- Adding new functionality to the shell etc.

1.3 PROBLEM STATEMENT

- Prone to costly errors, a single mistake can change the command which might be harmful
- Slow execution speed
- Design flaws within the language syntax or implementation
- Not well suited for large and complex task

1.4 OBJECTIVES

- The command and syntax are exactly the same as those directly entered in
- command line, so programmer do not need to switch to entirely different
- syntax: Writing shell scripts are much quicker
- Quick start

CHAPTER -2

LITERATURE SURVEY

MOTIVATION

If you haven't ever played Solitaire, Minesweeper, Hearts or FreeCell, it's safe to say you're in the minority. These simple Windows games have probably caused more lost worker hours than anything short of a worldwide coffee shortage. Whichever one was your favorite, the temptation to take just one more go at beating them—to get a faster time or a better score—was hard to ignore.

But as fun as these games were, they weren't actually designed for entertainment. At least not in their Windows incarnations.

The oldest of the four, Microsoft Solitaire, was first added to Windows 3.0 in 1990. Although the game (sometimes called "Patience") has existed since the late 1700s, this digital version seemed to be demonstrating that in the future we would no longer require a physical deck to play simple card games. But that's not what it was doing at all. Its real aim was far more modest: it was teaching mouse-fluency by stealth.

The intention was that Solitaire would get a generation of computer users still most familiar with a command-line input to teach themselves how to drag and drop, without realizing that's what they were doing. The fact that we're still dragging and dropping today suggests that it worked rather well.

Ramrao Adik Institute of Technology

CHAPTER -3

PROPOSED SYSTEM

3.1 INTRODUCTION OF PROPOSED SYSTEM AND ARCHITECTURE

The Operations are performed over the ubuntu operating system and there are various types of directories present in this system.

The main components here are:

Shopt: On Unix-like operating systems, **shopt** is a builtin **command** of the Bash shell which can enable or disable options for the current shell session

Tput: The **tput command** uses the terminfo database to make terminal-dependent information available to the shell. The **tput command** outputs a string if the attribute CapabilityName is of type string. The output string is an integer if the attribute is of type integer.

Read: **read** is a **command** found on **Unix** and **Unix**-like operating systems such as Linux. It reads a line of input from standard input or a file passed as an argument to its -u flag, and assigns it to a variable. In Linux based shells, like Bash, it is present as a shell built in function, and not as a separate executable file

esac: **esac** statement is to give an expression to evaluate and to execute several different statements based on the value of the expression. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.

echo: **echo command** in linux is used to display line of text/string that are passed as an argument . This is a built in **command** that is mostly used in shell scripts and batch files to output status text to the screen or a file. 2.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

The following mini-project of Minesweeper Game programmed using bash shell scripting language does not involve a conventional graphical user interface and thus it's specific requirements with their corresponding information is mentioned below

•   Shell can be accessed by user using a command line interface. A special program called Terminal in linux/macOS or Command Prompt in Windows OS is provided to type in the human readable commands such as "cat", "ls" etc. and then it is being execute. The result is then displayed on the terminal to the user. It will list all the files in current working directory in long listing format. Working with command line shell is bit difficult for the beginners because it's hard to memorize so many commands. It is very powerful, it allows user to store commands in a file and execute them together. This way any repetitive task can be easily automated. These files are usually called batch files in Windows and Shell Scripts in Linux/macOS systems

•A command-line interface (CLI) is an operating system shell that uses alphanumeric characters typed on a keyboard to provide instructions and data to the operating system, interactively. For example, a teletypewriter can send codes representing keystrokes to a command interpreter program running on the computer; the command interpreter parses the sequence of keystrokes and responds with an error message if it cannot recognize the sequence of characters, or it may carry out some other program action such as loading an application program, listing files, logging in a user and many others. Operating systems such as UNIX have a large variety of shell programs with different commands, syntax and capabilities. Some operating systems had only a single style of command interface; commodity operating systems such as MS-DOS came with a standard command interface but third-party interfaces were also often available, providing additional features or functions such as menuing or remote program execution.

CHAPTER –4

IMPLEMENTATION

4.1.SYSTEM BLOCK DIAGRAM
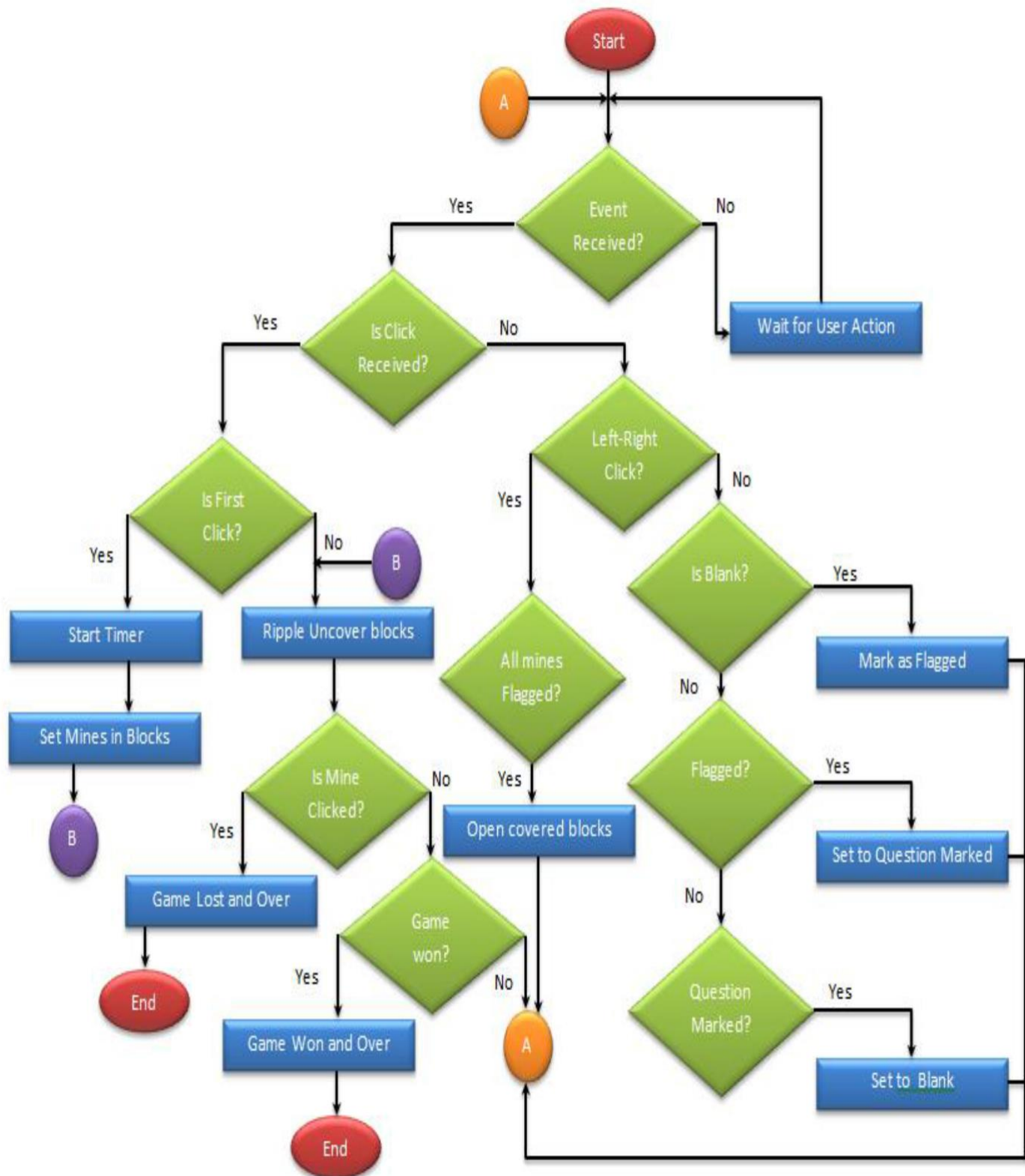
4.1 MODULE DESCRIPTION

The rules of the game

Since the game appeared in the 1960s, it has many variations and offshoots. Let us start having a look to the basic game and, then, we can have a look to which alternatives exist, even though it is almost impossible to make a complete list. And, with a bit of imagination, you can always invent your own version of the game!

The basic game

The Minesweeper is a single-player puzzle video game. At the beginning of the game, a 2D grid of identically looking tiles (or squares) is presented to the player. Some of them, unknown to the player, hide a mine (armed tile), some others not (safe tile). The number of armed tiles is known to the player and it determines the difficulty of the game. The goal of the game is to uncover all the tiles which do not contain a mine. If a tile hiding a mine is revealed, the player loses the game. If all the tiles not containing a mine are revealed, the player wins the game. Safe tiles contain a number signalling how many of the 8 adjacent tiles are armed1 Since revealing tiles next to a 0-tile cannot make the player lose the game, clicking on a 0-tile will make all the adjacent tile to be revealed as well. If in this process another 0-tile is hit, the process repeats over and over again. To avoid mistakes, the player has the possibility to mark tiles as armed, which implies they cannot be revealed (unless the label is removed). When the last safe tile is revealed, the game automatically marks all the remaining tiles as armed, while all the armed tiles are revealed when one of them is hit (usually indicating in some way the one hit by the player). Video games often give players the possibility to compete and so does the Minesweeper. The parameter to determine who did

the better job in two won games is time. This means that a timer starts when the first tile is revealed and stops when the game is ended. Typically it is shown and advances in real-time, just to add some excitement/stress to the game.

Variations and offshoots

The game itself opens to many possibilities and, indeed, many variations of it have been already implemented. Here below a list of different versions of the game.

> (i) A first alternative is to play not on a rectangular field. Any shape can be used, but tiles are always square.
>
> (ii) A similar idea is to change the shape of the tile. Triangles or hexagons are the most common alternative shapes. This reduces the maximum number of adjacent tiles.
>
> (iii) A more challenging option is to allow more mines to be under the same tile. Even if in principle not needed, usually a maximum number of mines allowed to stay together is fixed. Moreover, the

player is commonly told about how many tiles contain one mine, how many contain two, and so on.

(iv) Another rule that can be changed is the number of dimensions of the grid. It exists a 3D version of Minesweeper, which implies the possibility to have up to 26 adjacent mines per tile. Of course, the idea to have to safely walk on a mined field is a bit lost, but you could have to swim underwater in a dangerous sea with invisible poison localised here and there. . .1Tiles touching in only one point are also considered adjacent!

(v) Staying closer to the basic game, it is possible to impose periodic boundary conditions. This implies that all tiles have the same number of adjacent tiles and it forces the player to pay more attention close to the borders.

(vi) If you think about (or if you played the game), you will probably realise that it can happen to have to risk.

In principle the first choice always contains a non-zero probability to loose the game. A first variation of the original game is to rule out this possibility.

(vii) A much more trickier alternative is to rule out luck from the game. In this case, at no stage in the game, the player could loose because of bad luck. Said differently, the game will ensure that the entire grid can be fully deduced starting from the initial open space.

Clearly, some of the variants can be combined to make the game more interesting, or new ones could be introduced .Another aspect, which is not really a variant of the game but rather a help for the user, consists in allowing the player to reveal again an already revealed tile. This will make the game uncover all the adjacent unrevealed and not-marked-as-armed tiles, but, potentially, it can as well make the player loose. A safer variation of this idea is to reveal all the adjacent tiles only if as many adjacent tiles as the number of the chosen tile are marked as armed. It could be done iteratively by the game also on the tiles that are revealed after a tile has been chosen (as a cascade effect).

4.2 CODE

```bash
#!/bin/bash


    shopt -s extglob
    IFS=''

    piece=( $'\e[1;30m.' $'\e[1;34m1' $'\e[1;32m2' $'\e[1;35m3' $'\e[1;36m4' $'\e[1;31m5'
    $'\e[33m6' $'\e[1;37m7' $'\e[0;40;37m8' $'\e[0;40;37m#' $'\e[0;40;31mF'
    $'\e[0;40;33m?' $'\e[1;31m*' $'\e[0;40;31mx' )
    size=( 'S ' 10 10 15  'M ' 15 15 33  'L ' 20 20 60  'XL' 30 20 90 )

    function drawboard()
    {

      [[ "$dxt" ]] || { dxt=$mx; dyf=0; dyt=$my; }
```

```bash
      tput 'cup' $(( dyf+2 )) 0
      echo -n $'\e[40m'
      for ((j=dyf;j<dyt;j++)); do for ((i=0;i<dxt;i++)); do echo -n " ${piece[board[j*mx+i]]}";
done; echo ' '; done
      echo -n $'\e[0m'

      dxt=''

}

function newgame()
{

    #  n="$( expr index 'nNmM' "$1" )" # line kept as human readable version
     :( n='nNmM'; n="${n%$1*}"; n=${#n}

    mx=${size[n*4+1]}; my=${size[n*4+2]}; mb=${size[n*4+3]}; mf=0

     echo -n $'\e[0m'
     Clear
     echo 'Mine Sweeper version 1.1 august 2008 written by Feherke'
    echo "board : ${size[n*4]} size : $mx*$my mine : $mb flag : $mf
"$'\e[43;30m:)\e[0m'

     for ((i=0;i<mx*my;i++)); do bomb[i]=0; board[i]=9; done
     for ((i=0;i<mb;i++)); do while :; do r=$(( RANDOM%(mx*my) )); (( bomb[r] )) || break;
done; bomb[r]=1; done

     Drawboard
     echo $'<\e[1mh\e[0m/\e[1mj\e[0m/\e[1mk\e[0m/\e[1ml\e[0m> Move
<\e[1mg\e[0m> Step <\e[1mf\e[0m> Flag
<\e[1mn\e[0m/\e[1mN\e[0m/\e[1mm\e[0m/\e[1mM\e[0m> New <\e[1mq\e[0m> Quit'

     cx=0; cy=0
     status=1

}

function gameover()
{

     for ((i=0;i<mx;i++)); do for ((j=0;j<my;j++)); do
      (( bomb[j*mx+i]==1 && board[j*mx+i]==9 )) &&
board[j*mx+i]=12 Ramrao Adik Institute of Technology
```

```bash
    (( bomb[j*mx+i]==0 && board[j*mx+i]==10 )) &&
  board[j*mx+i]=13 done; done

  Drawboard
  tput 'cup' 1 52
  echo -n $'\e[43;30m:(\e[0m'

  status=0

}

function makestep()
{
  local i j
  local sx=${1:-$cx} sy=${2:-$cy}

  [[ "${board[sy*mx+sx]}" != @(9|10|11) ]] && return
  (( bomb[cy*mx+cx]==1 )) && { gameover; return; }

  [[ "$1" ]] || {
    dxt=$sx; dyf=$sy; dyt=$sy
    tput 'cup' 1 52
    echo -n $'\e[43;30m:o\e[0m'
  }

  (( dxt=dxt>sx?dxt:sx+1 )); (( dyf=dyf<sy?dyf:sy )); (( dyt=dyt>sy?dyt:sy+1 ))

  n=0
  for ((i=-1;i<=1;i++)); do for ((j=-1;j<=1;j++)); do
    (( (i!=0 || j!=0) && sx+i>=0 && sx+i<mx && sy+j>=0 && sy+j<my )) && ((
  bomb[(sy+j)*mx+(sx+i)]==1 )) && (( n++ ))
  done; done
  board[sy*mx+sx]=$n

  (( n )) || {
    for ((i=-1;i<=1;i++)); do for ((j=-1;j<=1;j++)); do
      (( (i!=0 || j!=0) && sx+i>=0 && sx+i<mx && sy+j>=0 && sy+j<my )) && makestep $((
  sx+i )) $(( sy+j ))
    done; done
  }

  [[ "$1" ]] || {
    Drawboard
    tput 'cup' 1 52
```

```bash
    echo -n $'\e[43;30m:)\e[0m'
  }

}

function putflag()
{

  [[ ${board[cy*mx+cx]} != @(9|10|11) ]] && return

  board[cy*mx+cx]=$(( (board[cy*mx+cx]-9+1)%3+9 ))

  (( board[cy*mx+cx]==10 )) && (( mf++ ))
  (( board[cy*mx+cx]==11 )) && (( mf-- ))

  (( mf==mb )) && {
   n=0
   for ((i=0;i<mx;i++)); do for ((j=0;j<my;j++)); do
     (( bomb[j*mx+i]==1 && board[j*mx+i]==10 )) && (( n++ ))
   done; done

   tput 'cup' 1 52
   echo -n $'\e[43;30mB)\e[0m'

   status=0
  }

  tput 'cup' 1 47
  echo -en "\e[0m$mf "

}


# /\/ /\ / /\/

newgame 'n'

while :; do

  tput 'cup' $(( cy+2 )) $(( cx*2 ))
  echo -en "\e[1;40;37m[${piece[board[cy*mx+cx]]}\e[1;37m]\b\b"

  read -s -n 1 a
```

```
[[ "$a" == '' ]] && { read -s -n 1 a; [[ "$a" == '[' ]] && read -s -n 1 a; }

echo -en "\b ${piece[board[cy*mx+cx]]} \b\b"

(( status!=1 )) && [[ "$a" != [nNmMrq] ]] && continue

case "$a" in
  'h'|'a'|'D'|'4') (( cx>0?cx--:0 )) ;;
  'j'|'s'|'B'|'2') (( cy<my-1?cy++:0 )) ;;
  'k'|'w'|'A'|'8') (( cy>0?cy--:0 )) ;;
  'l'|'d'|'C'|'6') (( cx<mx-1?cx++:0 )) ;;
  'g'|' '|'') makestep ;;
  'f'|'0') putflag ;;
  'n'|'N'|'m'|'M') newgame "$a" ;;
  'r') drawboard ;;
  'q') break ;;
  Esac

Done

echo -n $'\e[0m'
Clear
```
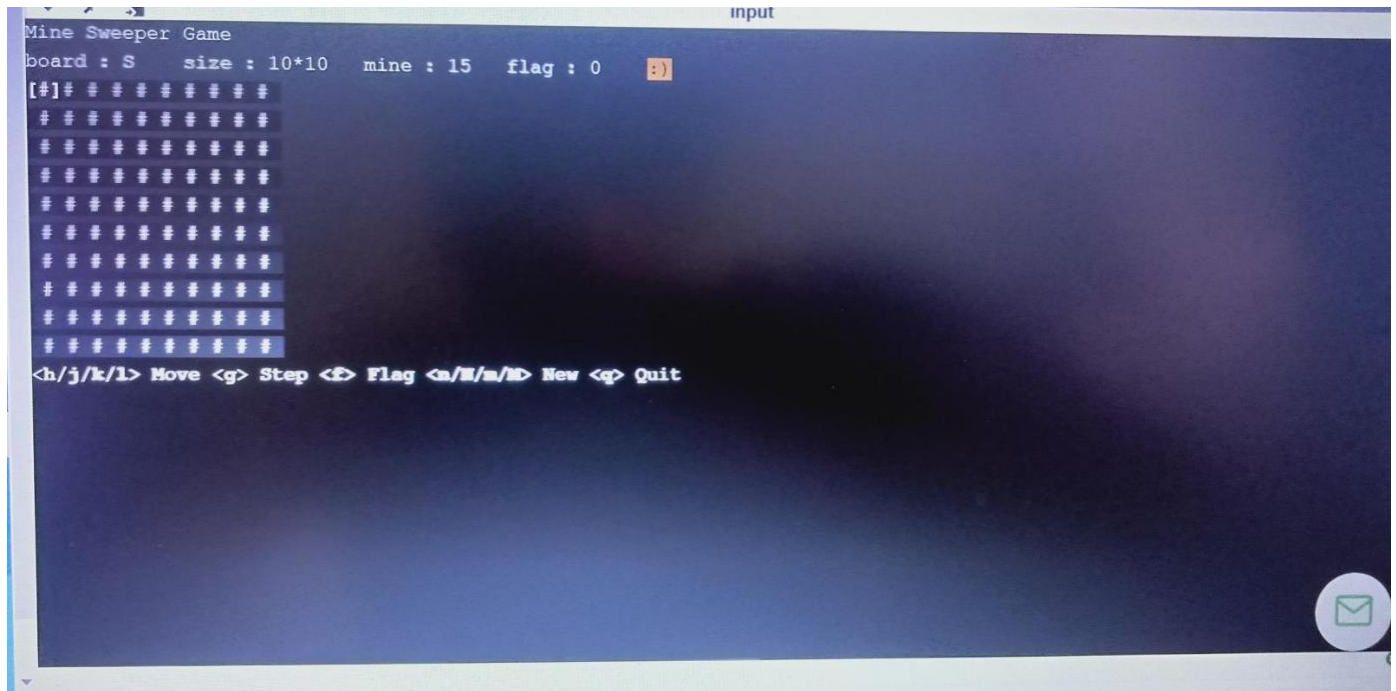
CHAPTER –5

RESULT

5.1.1 OUTPUT SNAPSHOTS
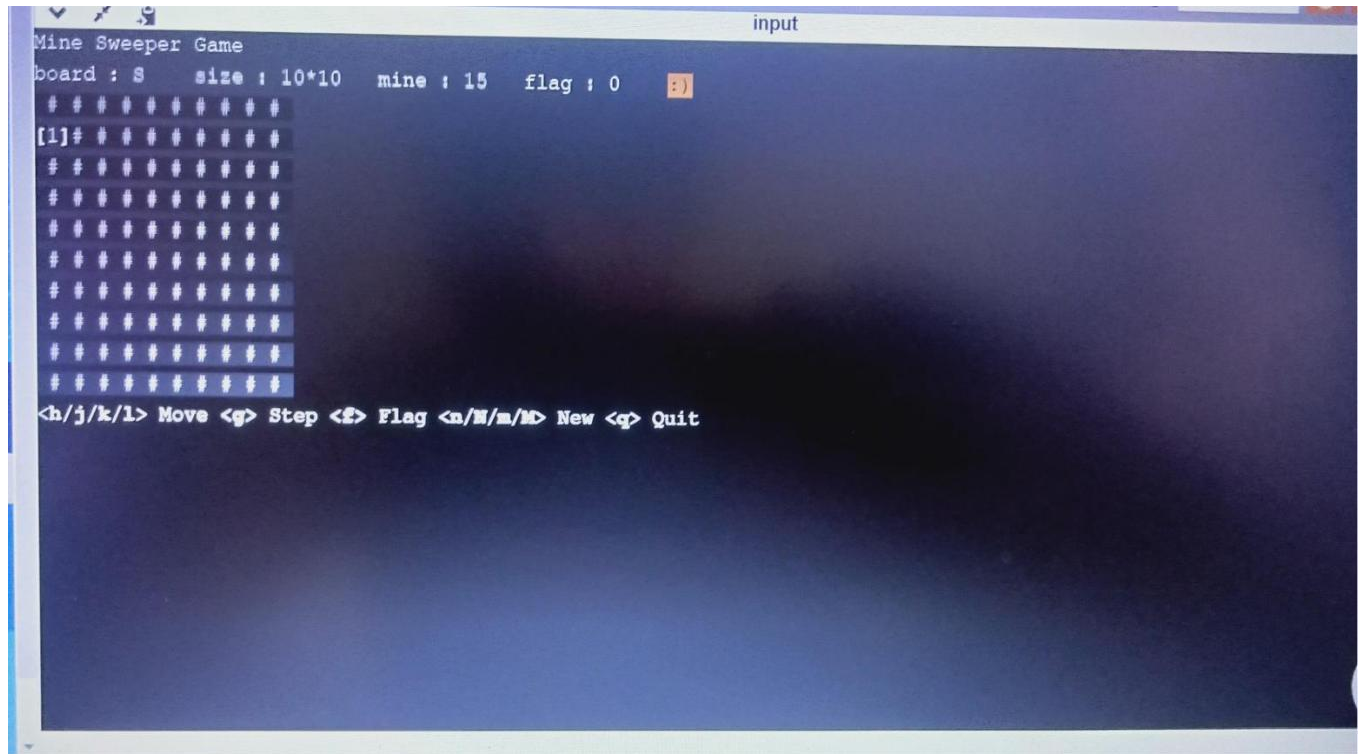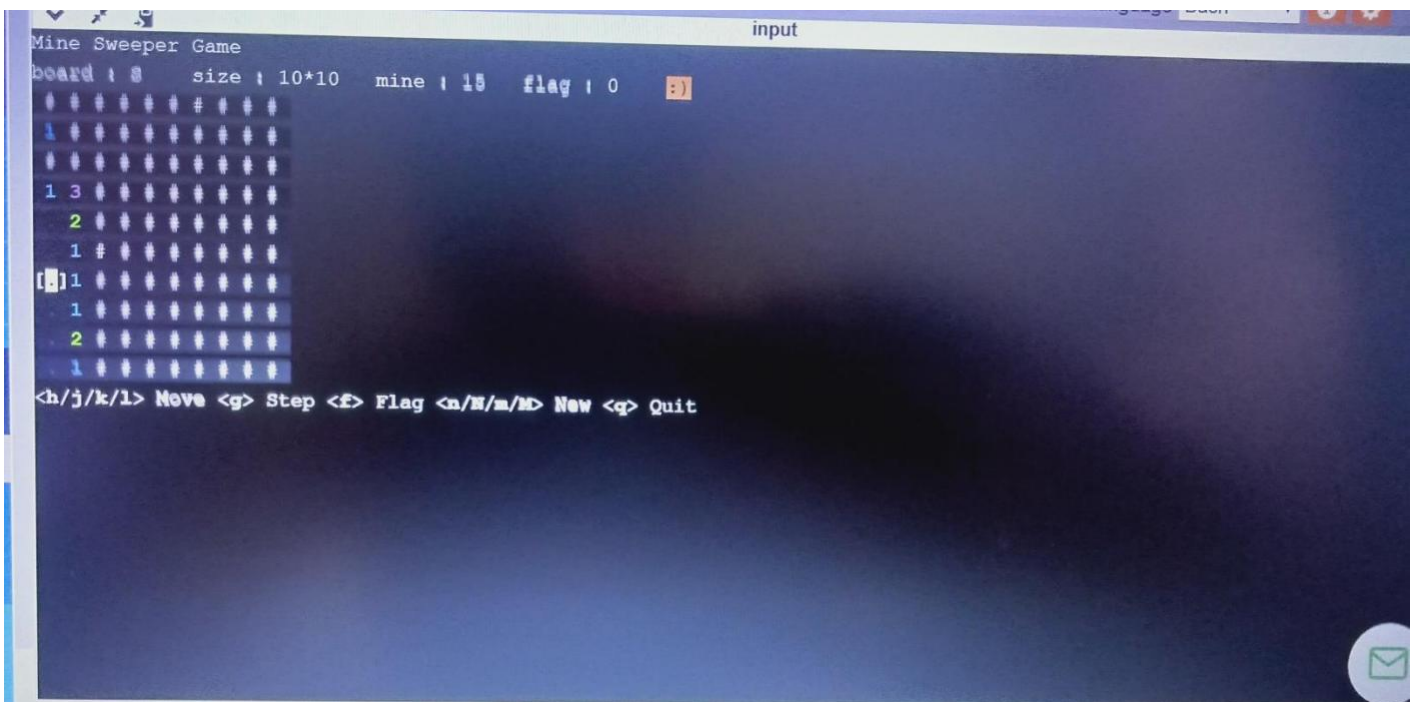
1.

2.

3.

```
Mine Sweeper Game
board : 8      size : 10*10    mine : 15    flag : 0    :)
# # # # # # # # # # #
1 # # # # # # # # # #
# # # # # # # # # # #
1 3 # # # # # # # # #
  2 # # # # # # # # #
  1 # # # # # # # # #
[.] 1 # # # # # # # # #
  1 # # # # # # # # #
  2 # # # # # # # # #
  1 # # # # # # # # #
<h/j/k/l> Move <g> Step <f> Flag <n/H/m/M> New <q> Quit
```

Ramrao Adik Institute of Technology

4.


```
                                                        input
Mine Sweeper Game
board : S      size : 10*10    mine : 15    flag : 0     :)
# # # # # # # # # #
1 # # # # # # # # #
# # # # # # # # # # #
1 3 # # # # # # # #
  2 # # # # # # # #
  1 # # # # # # # #
  1 # # # # # # #[1]
  1 # # # # # # # #
  2 # # # # # # # #
  1 # # # # # # # #
<h/j/k/l> Move <g> Step <f> Flag <n/N/m/M> New <q> Quit
```
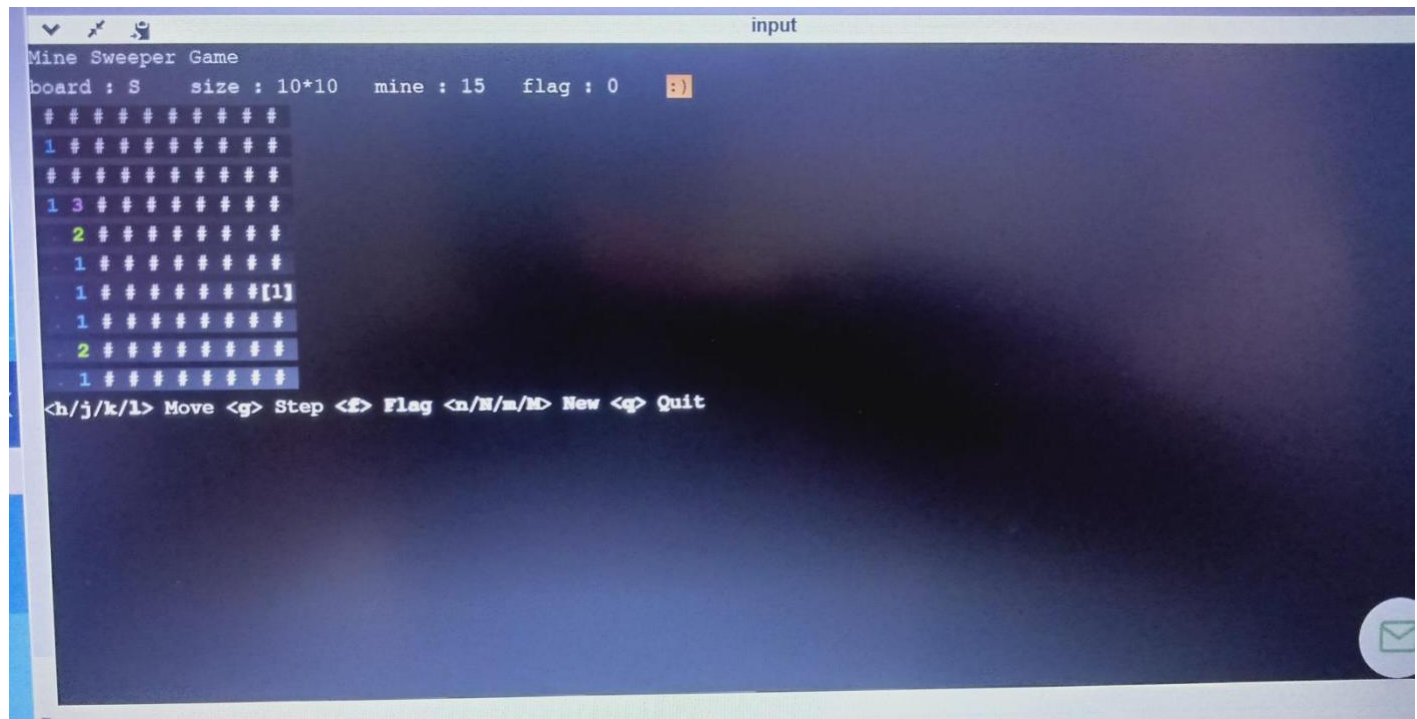
Ramrao Adik Institute of Technology

5.
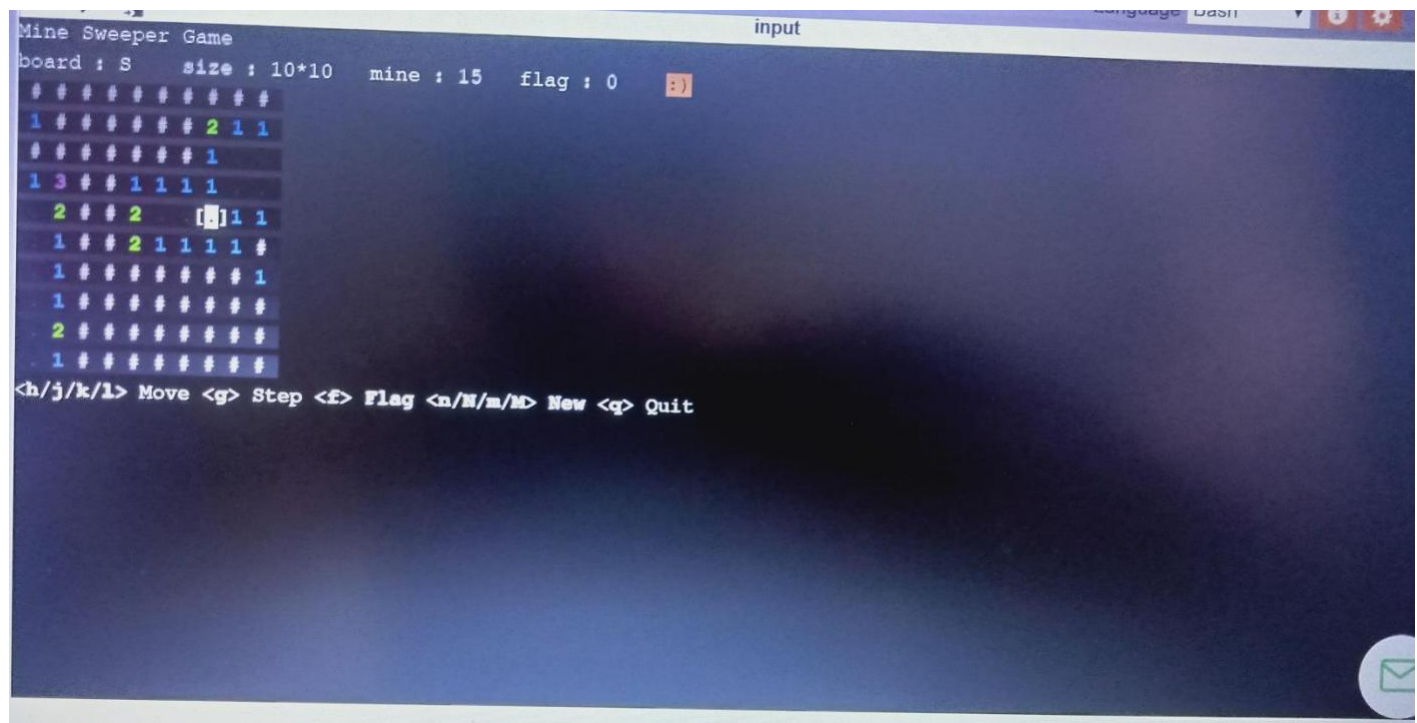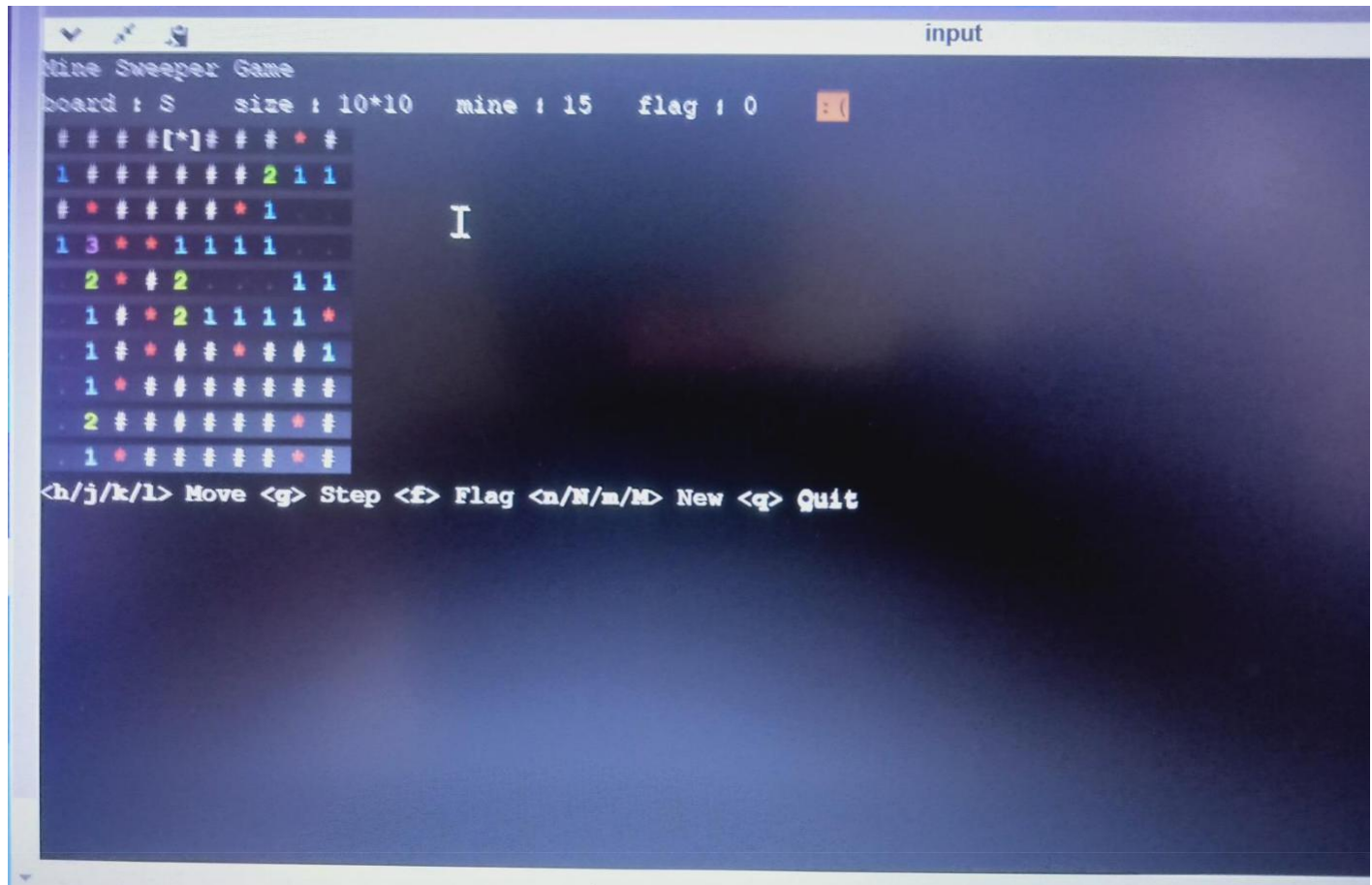
```
Mine Sweeper Game
board : S     size : 10*10    mine : 15    flag : 0    :)
# # # # # # # # # #
1 # # # # # # 2 1 1
# # # # # # # 1
1 3 # # 1 1 1 1
  2 # # 2      [.]1 1
  1 # # 2 1 1 1 1 #
  1 # # # # # # # 1
  1 # # # # # # # #
  2 # # # # # # # #
. 1 # # # # # # # #
<h/j/k/l> Move <g> Step <f> Flag <n/N/m/M> New <q> Quit
```

Ramrao Adik Institute of Technology

6.

Mine Sweeper Game
board : S   size : 10*10   mine : 15   flag : 0   :(

```
# # # #[*]# # # * #
1 # # # # # # 2 1 1
#  *  # # # # * 1
1 3 *  * 1 1 1 1
    2 * # 2        1 1
    1 # * 2 1 1 1 1 *
    1 # * # # * # # 1
    1 * # # # # # # #
    2 # # # # # # * #
    1 * # # # # # * #
```

<h/j/k/l> Move <g> Step <f> Flag <n/N/m/M> New <q> Quit

Ramrao Adik Institute of Technology

Testing and validation

Testing Details

Testing is a verification process for quality assessment and improvement. Testing is basically done to find errors, faults in the system. The basic goal of software development process is to produce the software that has very few or no errors

In an effort to detect errors soon after they are introduced each phase ends with verification activity such as reviews. However most of these verification activities in the early phase of the software development are based on human evaluation and cannot detect all the errors. Testing plays an important role in quality assurance for the software. It is a dynamic method for the verification and validation, where the system to be tested is executed and the behavior of the system is observed.

5.2.1 Black Box Testing

Black Box Testing is also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. Ramrao Adik Institute of Technology

5.2.2 White Box Testing

White Box Testing (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. White box testing is testing beyond the user interface and into the nitty-gritty of a system.

5.2.3 Unit Testing Unit

Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/super class, abstract class or derived/child class.

5.2.4 Integration Testing

Ramrao Adik Institute of Technology

Integration is the process by which components are aggregated to create larger components. Testing

the data flow or interface between two features is known as integration testing. Testing that occurs at lowest level is called unit/ module testing. As the units are tested and low level bugs are found and fixed, they are integrated and integration testing is performed against groups of modules. Integration Testing is also called as Structural Testing.

5.2.5 System Testing :

In system testing the behavior of whole system/product is tested as defined by the scope of the development project or product. It may include tests based on risks and/or requirement specifications, business process, use cases, or other high level descriptions of system behavior, interactions with the operating systems, and system resources. System testing is most often the final test to verify that the system to be delivered meets the specification and its purpose. System testing is carried out by specialist's testers or independent testers. System testing should investigate both functional and non-functional requirements of the testing.

5.2.6 Alpha Testing:

 Alpha testing is one of the most common software testing strategy used in software development. It's specially used by product development organizations. This test takes place at the developer's site. Developers observe the users and note problems. Alpha testing is typically performed by a group that is independent of the design team, but still within the company, e.g. in-house software test engineers, or software QA engineers. Alpha testing is final testing before the software is released to the general public. It has two phases:

 1. In the first phase of alpha testing, the software is tested by in-house developers. They use either debugger software, or hardware-assisted debuggers.The goal is to catch bugs quickly.
2. In the second phase of alpha testing, the software is handed over to the software QA staff, for additional testing in an environment that is similar to the intended use

CHAPTER –6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

In the end we would like to conclude that our aim was to make 'Minesweeper Game' using bash script. There must be more to it. It is not just only a simple and old computer game. – Minesweeper has become a part of Windows; it is for computer games (almost) as legendary as Tetris or Pong are for video games in general. Minesweeper is definitely more than just a computer game.

6.2 FUTURE SCOPE

After certain tweaks in the mini project it can be made more visually appealing at the same time it would have immense potential if taken online where people will be able enjoy the game .

6.3 REFERENCES

- **www.google.com**
- **www.wikipedia.org**
- **www.studycircle.org**
- **www.tutorialpoint.com**
- **www.w3schools.com**
- **www.youtube.com**
- **www.academia.edu**