UNIX PROJECT(1)

Submitted to:

Ms.Deepthi

Assistant Lecturer

Department of Information Technology

NITK Surathkal

Ms.Sangeeta

Assistant Lecturer

Department of Information Technology

NITK Surathkal

Akshay Pandita - 16IT151
Nimish Mangal - 16IT233
Shaswat Kumar Patel - 16IT243
Dashan Jot Singh - 16IT216
N. Nishanth - 16IT229

Date-26th November,2017

Declaration(2):

This is to certify that the B.Tech Mini project entitled “**UNIX MINI PROJECT” on ‘MINESWEEPER’** submitted by :

```
Akshay Pandita  – 16IT151
Nimish Mangal – 16IT233
Shaswat Kumar Patel – 16IT243
Dashan Jot Singh – 16IT216
N. Nishanth – 16IT229
```

as the record of the work carried out by them is accepted as the B.Tech Mini project submission in partial fulfillment of the requirements for mandatory learning course of IT202, Unix Programming and Practice in the Department of Information Technology.

Date: 26th November 2017.

Place: NITK Surathkal.

Signature and Seal

Abstract(3):

The game is **MINESWEEPER** and is made using BASH Shell Scripting.

Here the code is divided in 3 parts :-

- ***Constructing Boards and Putting of Mines(3.1):-*** Using 2-D array and using a Random function that we will create board and place mines at random position using a probability of merely 0.01 . This ensures that the mines are placed sufficiently on a random basis. If we reduce the probability of the condition of the mines to be placed it may happen that mines are placed close to each other on the minesweeper board.

- ***Input from user and Searching(3.2):-*** Input from user is taken and corresponding element is searched if the position taken by the user as an input corresponds to the place of a mine then it leads to the bombing of that place and hence the game ends otherwise if the position input from the user doesn't correspond to the place of the mine we then check for the number of mines in it vicinity and hence we display the number of bombs in its vicinity . For Example – If there are 2 bombs in its corresponding 8 neighboring cells then the position will show 2 otherwise similarly will show 1 or . respectively .

- ***Deciding(3.3): -*** If the user inputs all the positions that don't correspond to the places of the mines then only he wins otherwise as soon as the user hits a mine position the game ends . For this to occur we check whether the respective player satisfies the winning condition or not. If the user satisfies the winning condition then only the user wins otherwise he will lose.

The law of the game includes taking the input coordinate from the keyboard entered by player and then corresponding board cell will be raveled and if the player is the hit then game will be terminated and if all till end player doesn't hit the mine then at end the player will won.

CONTENTS(4):

Introduction(5) :

This game is a take on the classic game MineSweeper. The game places the player in minefield where the objective is to avoid all the mines laid out under the tiles. The mines are triggered when the player uncovers them. The mines are never in the same place, this ensures that every game is a new experience. This is strategy game where the player has to find the patterns and use logical reasoning to get himself out of this mine field. The game operates on a co-ordinate input system.

The only win scenario in this game is when the player uncovers all the tiles except the ones covering the mines and the player will be awarded his freedom from the minefield.

The game uses shell script and uses many of the useful commands such as shuf command, the command used to give a random number in a given range and declare command, the command used to declare a 2d array in shell. Some interesting features of shell have been used like changing the color code to display different colors on the terminal. The goal of the project is to gain proficiency in shell scripting, hence the project was coded in shell script. Since the code was made in shell script it also gave us a good opportunity to gain some insight of the bash and the shell script and also increase our thereby limited knowledge by knowing more about the different functions available only in the shell script.

CHALLENGES(5.1):-

- The first challenge we faced was to design the algorithm for the game MINESWEEPER. It was quite a challenge for us because of the pure logic involved in this game. We had to compute the number of bombs in a cell's vicinity and we did this using the for loop and accessed all the nearby positions and checked whether they had bomb or not. Then we had to use a randomly generated number function which is SHUF.

- The second challenge we faced was to convert this algorithm to the shell script code and use the syntax for every distinct function in bash. This was not only a challenge for us but also it provided a motivation to us to get better knowledge about the different bash functions.

- The third challenge for us to give different colours to the game to make it more attractive . We did so using the different colour codes and the background codes used in bash and shell script.

MOTIVATION(5.2):-

- The first motivation that worked for us was to get to know more about the different functions in shell script and to know more about their syntax thereby increasing our knowledge that was previously limited in bash and shell script.

- The other and the main motivation for us to do this project was provided by our UNIX professors **MS. DEEPTHI** and **MS. SANGEETA.**

METHODOLOGY (6): -

6.1 SYSTEM REQUIREMENTS

- FOR WINDOWS OS
    1. Putty
    2. Cygwin
    3. Command prompt

- FOR UBUNTU OS
    1.Terminal
    2.VI editor
    3.Sublime Text

ALGORITHM(6.2):-

Step-1:

Our project involve drawing of two board first one is table array(named as table) and second one is discovered array(named as discovered) so we use 2-D array for that we used **declare -A** command.

Step-2 :

We now create the board and place the mines in it using the function constructtable**().** In this function for placing mines at random position we use **shuf** command with the probability of 1% and we use 12 mines in the whole game. If 12 mines are not placed using random function then the loop again iterate.

Step-3 :

Next we use **constructtable()** function to update the board after each turn it will search for the mines in surrounding 8 squares if there is a mine then it will increase the mine no of surrounding 8 shell by one.

Step-4 :

Next we use **viewtable()** function to display the board after each turn input coordinate in fore become one and corresponding coordinate in board is revealed using echo –ne command which display in same line and ([1;34m ${table[$i,$j]}) use to display the color.

<u>Step-5 :</u>

Next in this code we uses the **start().**

This function again refresh the board and put mines 12 and make the every cell of fore and board equal to zero.

The **Gameover()** function will terminate the game . if in the bore cell contain mine corresponding to the input coordinate then it will call the display function and terminate the game.

<u>Step-6 :</u>

Wingame() iterate throughout the board and check for the mine after each turn if mine is there then it will call the **gameover()** function and terminate the game. Else if all cell are uncovered then it display the message that player has won the game.

Step-7 :

This is the main step in the algorithm of the game in this it will take the x, y coordinate as input from the **newgame()** function and check whether the corresponding board coordinate if those are zero then it will call Findadjacentbubble**()** function which reveals the 8 neighbours of given input

If any of 8 input is zero then the chain function will call the BubbleDisplay() which Scans the board if there are any zeros with covered neighbours and reveals then then it scans the whole board again.

Step-8 :

From main method, if user input **'y' or 'Y'** we call the **instructions()** method which display rules of the game then we call the new game function

In **newgame()** function we will take the input coordinate from the user if x > -1 and y >-1 it will call the **Findbubblepoint()** function if table coordinate is -1 then it will call **gameover()** method else it will display the result.

IMPLEMENTATION(7)

7.1 LIST OF MAIN UNIX COMMANDS

- SHUF
- CLEAR
- ECHO
- FOR LOOPS
- SELECT LOOP
- IF STATEMENTS
- DECLARE
- CASE
- DATE

7.1.1 USER DEFINED FUNCTION

• ConstructTable()	- To setup a new minefield
• ViewTable()	- To display the board after each move
• Start()	- To reset game variables to initial values
• Gameover()	- Display losing Scenario
• Wingame()	- Display winning Scenario
• CheatWin()	- Alternate to winning Screen (mine mania)
• FindBubblePointt()	- Initiates the changes to the board after each move
• FindAdjacentBubble()	- Reveals the surrounding 8 neighbours of given tile
• BubbleDisplay()	- Reveals the chain of zeroes and their neighbours
• Rules()	- Instructions screen
• newgame()	- To begin a new game
• endPage()	- End screen
• Time()	-Time taken to complete the Game

7.2 USES AND SYNTAX OF COMMANDS

1. SHUF(shuf   -i 0-10(RANGE) -n 1):-- to get a random value in the range given.

2. CLEAR(clear):--used to clear the terminal screen so that the execution of the program can begin from the top of the screen.

3. ECHO (echo -e “\t\t Hi! There\n” or echo -n “Hi! There”):--to print a string. -e is used to give specific function to some characters and -n is used to print string on the same line.

4. DATE(date +”%H%M%S”):--used to give the present hour , minutes and seconds.

5. DECLARE(declare -A board):-- the declare command is used to create an array be it 1-D, 2-D and even 3-D matrices.

6. IF (if (statement to be checked followed by then and then terminated by fi))
   (if [$n -gt 10]
   then
   echo “Number is greater than 10 “
   fi) it is used to do a block of commands if any condition is satisfied.

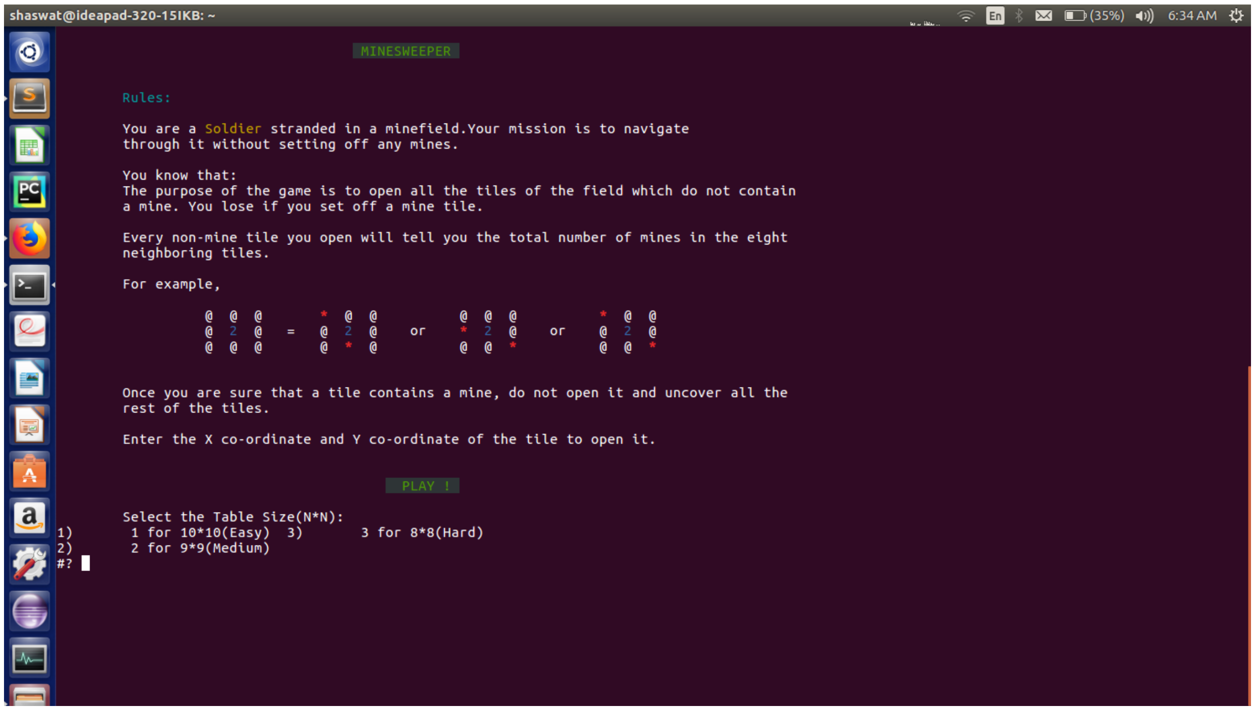
7. CASE (case $option in (then followed by a certain number of possible options of $option)) this command is basically used in the running of a select loop a certain number of times only .

8. FOR (for((i=0;i<10;i++))
   ```
   do
   echo “$i”
   done)
   ```
   this command is used in shell script to repeat a specific block of statements a specific number of times only(in this case a specific block of statements is executed 10 times).

9. SELECT (select sz in "   1 for 6*6" "   2 for 8*8" "   3 for 10*10") this command is used to provide a list of options that can be chosen and specific statement can be executed when any one option is chosen.

RESULT AND ANALYSIS (8):

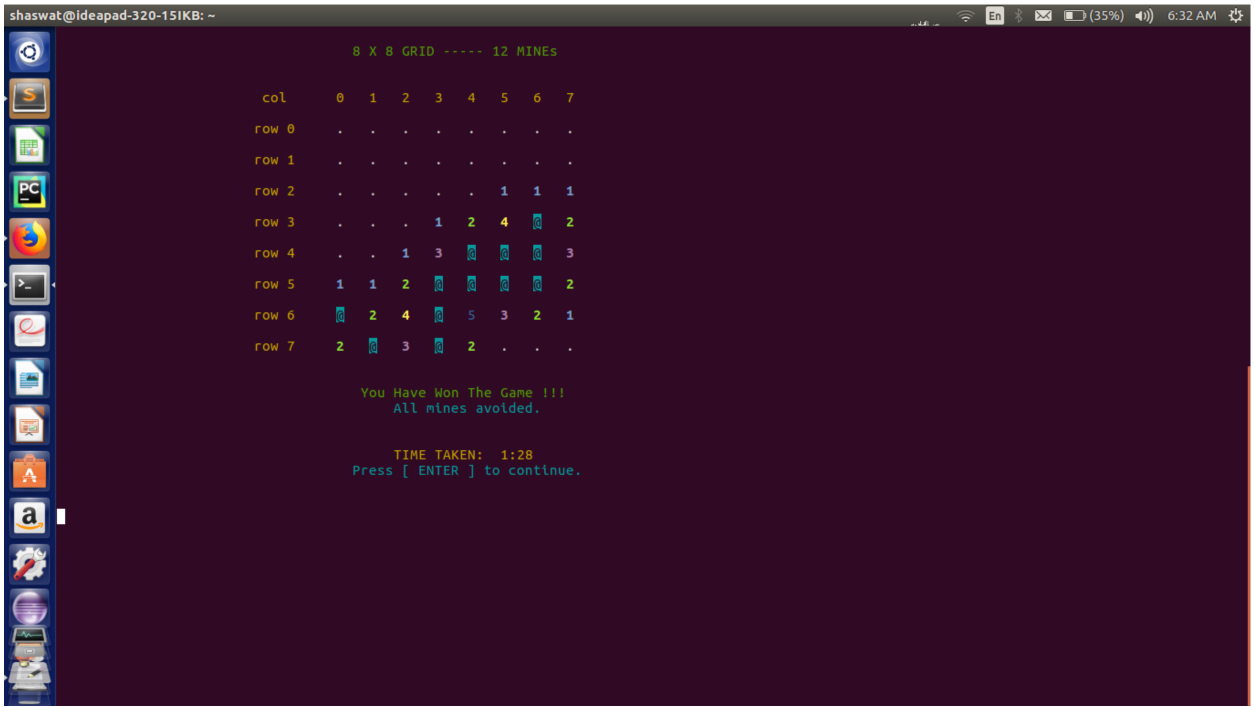
The initial output screen. This is the initial screen that will be showed up when the user runs the game MINESWEEPER file (with the .sh extension)This screen also includes the instruction of the game.



```
shaswat@ideapad-320-15IKB: ~                                    (35%)  6:34 AM

                                        MINESWEEPER


         Rules:

         You are a Soldier stranded in a minefield.Your mission is to navigate
         through it without setting off any mines.

         You know that:
         The purpose of the game is to open all the tiles of the field which do not contain
         a mine. You lose if you set off a mine tile.

         Every non-mine tile you open will tell you the total number of mines in the eight
         neighboring tiles.

         For example,

                    @  @  @       *  @  @          @  @  @          *  @  @
                    @  2  @   =   @  2  @    or    *  2  @    or    @  2  @
                    @  @  @       @  *  @          @  @  *          @  @  *


         Once you are sure that a tile contains a mine, do not open it and uncover all the
         rest of the tiles.

         Enter the X co-ordinate and Y co-ordinate of the tile to open it.


                                             PLAY !

         Select the Table Size(N*N):
1)        1 for 10*10(Easy)  3)       3 for 8*8(Hard)
2)        2 for 9*9(Medium)
#? 
```

FIGURE 1

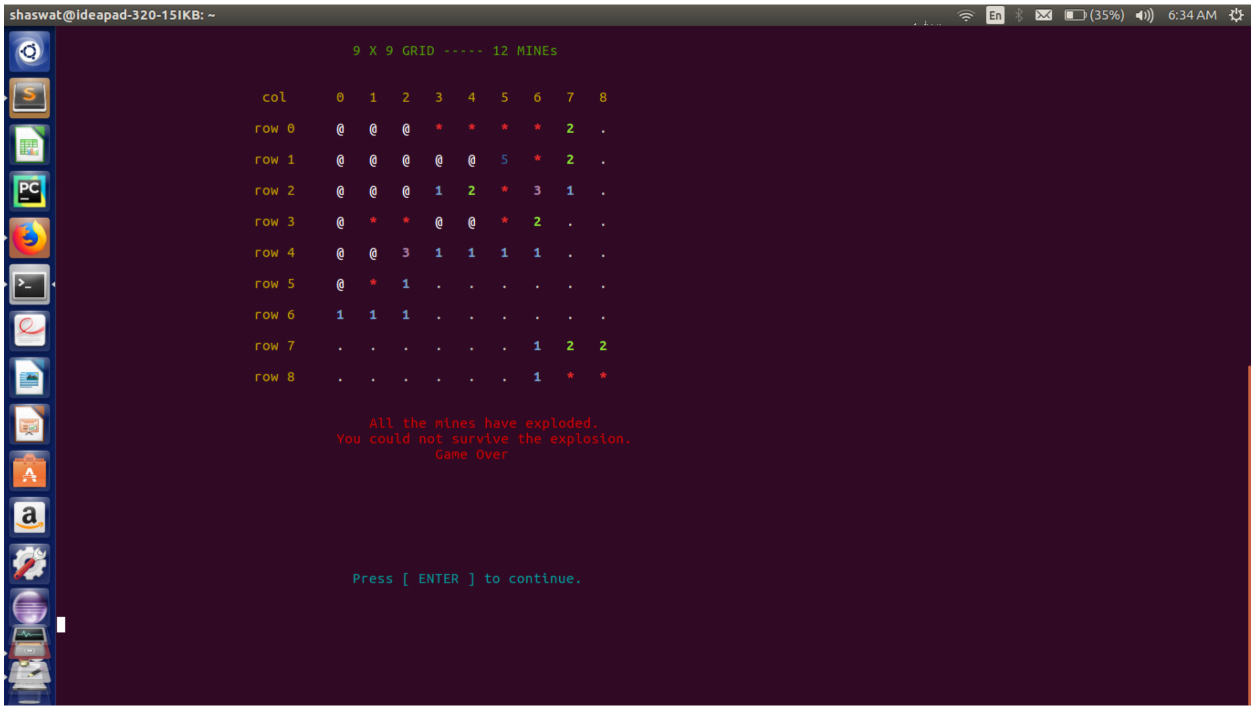
The intermediate screen where the adjacent zeroes are shown. This screen shows the immediate number of bombs linked with each of the position. 2 means that there are exactly 2 bombs in its nearby 8 cells i.e. 2 on top , 2 on right , 2 on left and 2 down.



```
shaswat@ideapad-320-15IKB: ~

                      8 X 8 GRID ----- 12 MINEs

          col       0   1   2   3   4   5   6   7

         row 0      .   .   .   .   .   .   .   .

         row 1      .   .   .   .   .   .   .   .

         row 2      .   .   .   .   .   1   1   1

         row 3      .   .   .   1   2   4   @   2

         row 4      .   .   1   3   @   @   @   3

         row 5      1   1   2   @   @   @   @   2

         row 6      @   2   4   @   5   3   2   1

         row 7      2   @   3   @   2   .   .   .


                       You Have Won The Game !!!
                           All mines avoided.


                           TIME TAKEN:  1:28
                      Press [ ENTER ] to continue.
```

FIGURE 2

Finally, the endgame screen. This screen shows the end of the game when the user hits a position of a mine which were randomly placed by the shuf function. If the user never hits the position of a mine then he won’t lose i.e. he will win.



```
shaswat@ideapad-320-15IKB: ~

              9 X 9 GRID ----- 12 MINEs

   col       0   1   2   3   4   5   6   7   8

  row 0      @   @   @   *   *   *   *   2   .

  row 1      @   @   @   @   @   5   *   2   .

  row 2      @   @   @   1   2   *   3   1   .

  row 3      @   *   *   @   @   *   2   .   .

  row 4      @   @   3   1   1   1   1   .   .

  row 5      @   *   1   .   .   .   .   .   .

  row 6      1   1   1   .   .   .   .   .   .

  row 7      .   .   .   .   .   .   1   2   2

  row 8      .   .   .   .   .   .   1   *   *

                 All the mines have exploded.
             You could not survive the explosion.
                         Game Over

              Press [ ENTER ] to continue.
```

FIGURE 3

CONCLUSION(9)

The game aimed at implementing the classic game MineSweeper using the knowledge gained in the
subject shell scripting. The game is an interactive co-ordinate based game which involves a lot of logical reasoning. The many commands used in the code are a product of research on the topic as well as knowledge taught in class. New commands were learnt and implemented in the program such
as shuff command, the command used to give a random number in a given range and declare command,
the command used to declare a 2d array in shell. The usage of functions was seen throughout the program, the functions played a major part in the program.

The commands used:

clear	- Clearing the terminal
echo	- Printing various elements
declare	- Declaring Arrays
for loops	- Various loops
if statements	- Various conditions

References(10):

https://linux.die.net/man/1/shuf
https://linux.die.net/abs-guide/declareref.html
books:--
UNDERSTANDING UNIX BY SUMITABHA DAS
Github.co.in
Wikipedia.co.in