Sudoku Application– Includes Smart Assist and Solver

Sarath Vadakkepat
Vigneshwar Shankar

Acknowledgement

I would sincerely like to thank my parents and friends who encouraged meand helped me to complete this project successfully.

I sincerely express my heartfelt gratitude to Sujith Vadakkepat, Prahlad Suresh, Prachet Verma, Aditya Gautam, Vigneshwar Shankar and everyone for their encouragement and support.

Index

Preface

Sudoku is one of the most famous game played by billions of people across the world. Sudoku is a logic based, combinational, number placed puzzle. The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 sub-grids that compose the grid (also called "boxes", "blocks", "regions", or "sub-squares") contains all of the digits from 1 to 9. The puzzle setter provides a partially completed grid, which typically has a unique solution.

The puzzle was popularized in 1986 by the Japanese puzzle company Nikoli, under the name Sudoku, meaning *single number*. It became an international hit in 2005. Although the 9×9 grid with 3×3 regions is by far the most common, many variations exist. Sample puzzles can be 4×4 grids with 2×2 regions; 5×5 grids with pentomino regions have been published under the name *Logi-5*

Android is one of the widely used mobile operating system currently. The love for Sudoku will make the enthusiasts find a medium in which they can play it as they wish. Hence an android application for playing Sudoku will be an ideal choice.

Mathematics of Sudoku

A completed Sudoku grid is a special type of Latin square with the additional property of no repeated values in any of the 9 blocks of contiguous 3×3 cells. The relationship between the two theories is now completely known, after it was proven that a first-order formula that does not mention blocks is valid for Sudoku if and only if it is valid for Latin Squares.

The number of classic 9×9 Sudoku solution grids is 6,670,903,752,021,072,936,960 (sequence A107739 in OEIS), or approximately 6.67×10^{21}. This is roughly 1.2×10^{-6} times the number of 9×9 Latin squares. Various other grid sizes have also been enumerated—see the main article for details. The number of essentially different solutions, when symmetries such as rotation, reflection, permutation and relabelling are taken into account, was shown to be just 5,472,730,538 (sequence A109741 in OEIS).

The maximum number of givens provided while still not rendering a unique solution is four short of a full grid (77); if two instances of two numbers each are missing from cells which occupy the corners of an orthogonal rectangle, and exactly two of these cells are within one region, there are two ways the numbers can be assigned. Since this applies to Latin squares in general, most variants of Sudoku have the same maximum. The inverse problem—the fewest givens that render a solution unique—was recently proven to be 17. A number of valid puzzles with 17 givens have been found for the standard variation without a symmetry constraint, by Japanese puzzle enthusiasts, and 18 with the givens in rotationally symmetric cells. Over 48,000 examples of Sudoku puzzles with 17 givens resulting in a unique solution are known.

In 2010 mathematicians of the University of Southern California showed that the arrangement of numbers in Sudoku puzzles have greater Shannon entropy than the number arrangements in randomly generated 9×9 matrices. This is because the rules of Sudoku exclude some random arrangements that have an innate symmetry.

APPLICATION DETAILS

Name	: Sudoku
Platform	: Android
APK name	: Sudoku_SV17
	: Sudoku_SV17v3
	: Sudoku_SV17v47
Versions Supported	: v2.3 upwards
Screen Sizes	: 3.4” to 4” – Sudoku_SV17
	: 3.2” to 3.4” – Sudoku_SV17v3
	: 4.7” - Sudoku_SV17v47
Developed in	: Eclipse Galileo with ADT plugin
Key Features	: Solver
	: Smart Assist
Devices Tested on	: HTC Desire S
	: HTC Wildfire S
	: Samsung Galaxy Ace
	: Sony Ericsson Xperia Live with Walkman
Developer (Code)	: Sarath Vadakkepat
Developer (Graphics)	: Vigneshwar Shankar
Documentation	: Yes
Size (APK)	: 893 KB (0.872 MB)
Sensors Used	: Touch Screen

Permissions	: None other than default
Build SDK	: 10
Targer SDK	: 10 and above
ScreenShots	: Yes
Size when installed	: 1.07 MB
Download Links	:
	:
	:

Application Introduction

Android is one of the most widely used mobile operating system along with Windows Phone 7 and the iOS. More than 50% of the smart phones present in the world currently use Android OS. Sudoku is a game played by millions of people. Sudoku games are available for play in newspapers and there are many websites online which facilitate the players to play online.

Hence, it is ideal and convienient for people if they have a Sudoku application installed in their phone. Thus the users can play the game whenever they wish to and also overcomes problems that exist due to dependence on the newspapers and websites. This application was developed by keeping in mind the general interest of the users and ease. Also, to assist the user there is an option of “Smart Assist” available. The user can also choose to get the solution of a Sudoku table by using the “Solver”.

The application initially was made just to have a Sudoku solver but in due course of time, playing option was added to it. The graphics was custom designed for this and this application is expected to be released as a free of cost application to all users.

The key features and the novelty of this application includes

1. Sudoku Solver
2. Smart Assist
3. Custom NumBoard
4. Customised and Efficient Game Engine

APPLICATION ARCHITECTURE

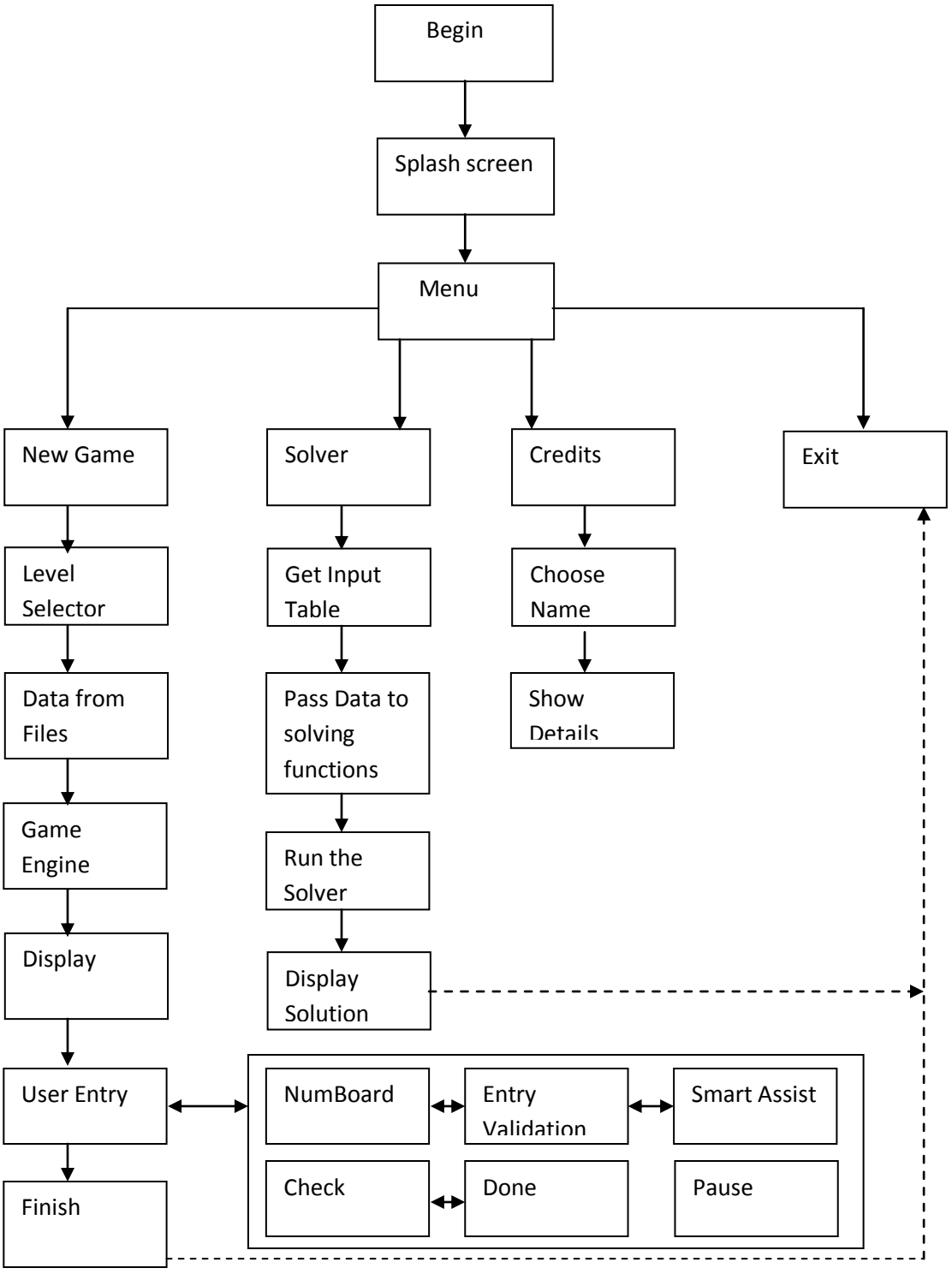


Fig. 1

GAME ENGINE

The Sudoku puzzles that are used for playing are generated by the game engine after manipulation of data samples. The data samples for each difficulty level i.e. Easy, Medium and Hard and stored in text files. The game engine selects one of the puzzles and sends it through the manipulation architecture that generates another version of the same. The game engine is extremely optimized and highly efficient coding dynamics put to test. The game engine was completely conceptualized and designed by myself. The code was given much iteration to completely optimize its functionality and check for accuracy.

Algorithm

1. Pick one data from the sample space according to the level of difficulty as chosen by the user.
2. Initiate random processes to determine the level of randomizations needed.
3. Send data through a series of manipulation according to the aforesaid random process and also the random process of each level.
4. Data shifted by rows and colums and orientations changed depending on the random process that is acting on that part.
5. Data then comes to a permuted form after passing through 6-step random process.
6. Output the final puzzle as a string output of 81 characters.

Flowchart

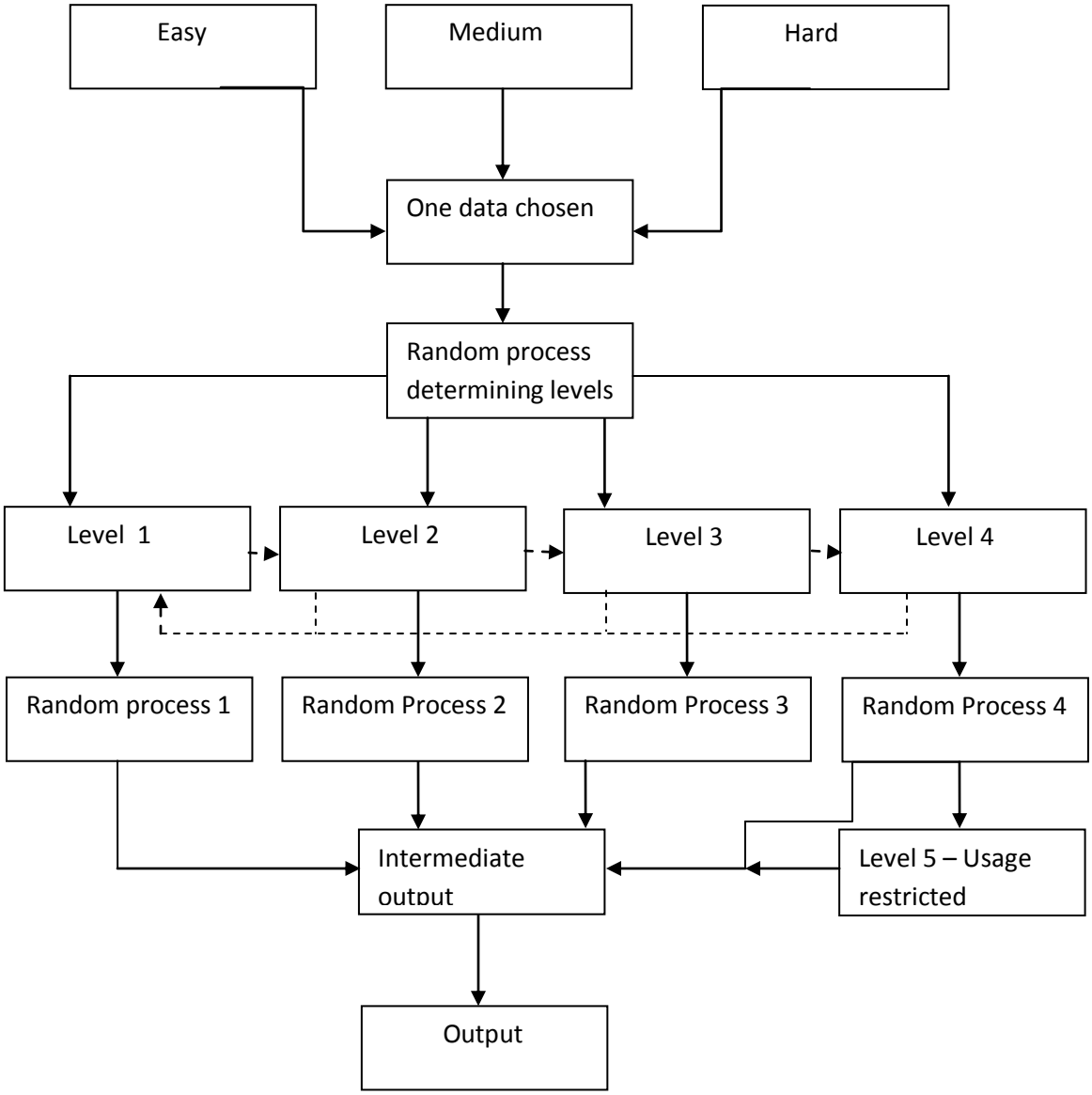


Fig. 2

The different types of shuffling include

1. Level 1
    1. Normal
    2. Clockwise Rotation
    3. Clockwise Rotation twice
    4. Anti-Clockwise Rotation
2. Level 2
    1. Level 1
    2. Mirrored Horizontal
    3. Mirrored Vertical
3. Level 3
    1. Level 2
    2. Shift Vertical Block 1, 2, 3 and Horizontal Block 1, 2, 3 in one of 36 orientations.
4. Level 4
    1. Level 3
    2. Mirrored Vertical
    3. Mirrored Horizontal
    4. Supercharged level, Sub Vertical Row 1,2,3 and Sub Horizontal Column oriented in one of 36 ways.

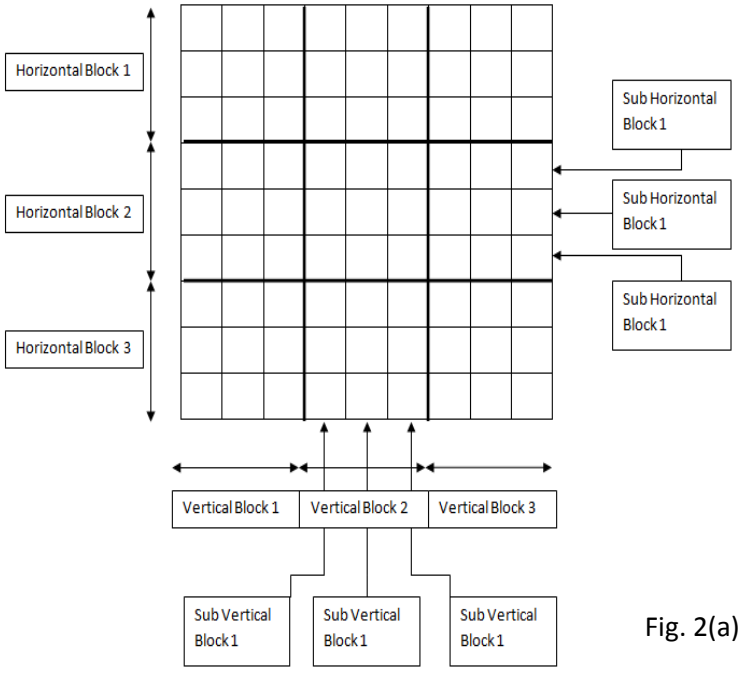


Fig. 2(a)

NumBoard

The interface that facilitates user entry is not based on the default keyboard provided by the android operating system. For the application a new keyboard was designed. Titled “NumBoard” prominently is displayed as a pop-up when the user has to enter any number rom 1-9 and it also features erase option also.

Fig. 3

Algorithm

1. Listen for clicks on cells
2. Verify is Smart Assist is turned
3. Display the NumBoard appropriately.
4. Dismiss on user press

FlowChart

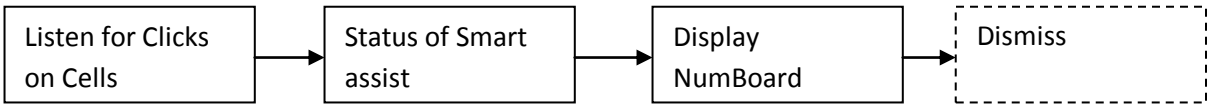


Fig. 3(a)

Smart Assist

The feature of Smart Assist included in this application is one of the novelty achieved in this application. This feature is aimed in providing help to the users while the user is unable to proceed the game.

The idea of this feature is to simply tell the user which of the numbers are possible to be inserted in a cell depending on the dynamic situations present in the game. This feature can be turned on my checking the “Assist” option available during gameplay. When assist is turned off, the clicking on cell would project the whole NumBoard consisting of all the number from 1-9, but on turning the assist on, clicking on any cell would display the numbers possible to that particular cell thereby disabling the other buttons.

The screen shots depicting this feature are shown below.

Without Assist

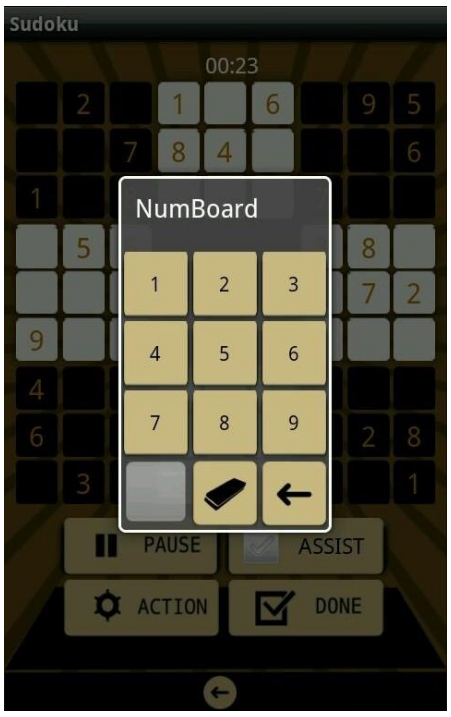


Fig. 4

With Assist

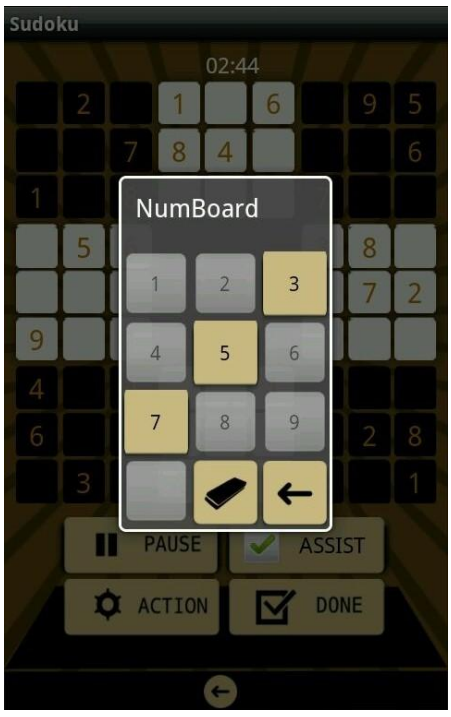


Fig. 4(a)

Algorithm

1. Listen for clicks on cells
2. Status of the “Assist” option
3. If assist it activated, function for checking row, column and cube is executed.
4. Display Set X of numbers where Set X is subset of {1,2,3,4,5,6,7,8,9}
5. Displays the NumBoard with Set X.

Flowchart

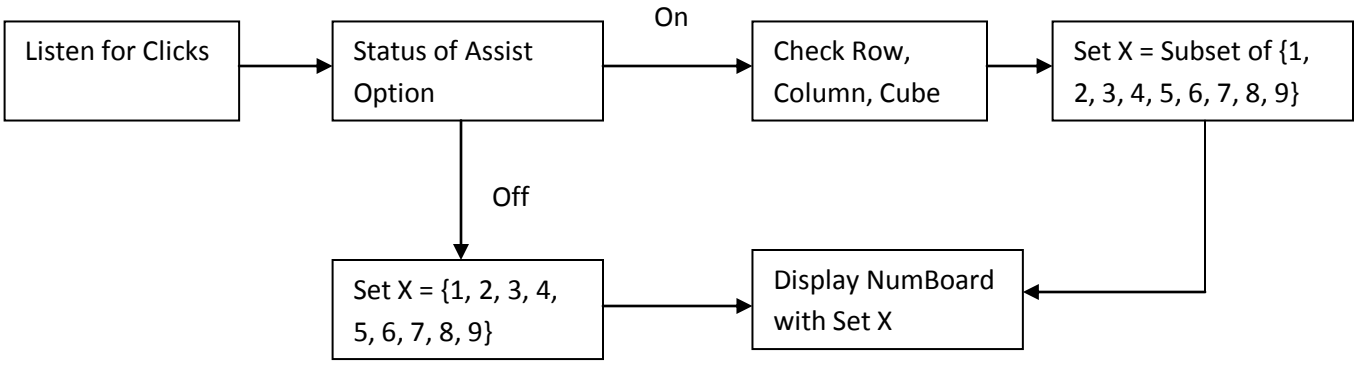


Fig. 4(b)

The New game option enables the user to start a new game. The user is then given a option to choose the level of hardness i.e. Easy, Medium, Hard.

Once the user has clicked on his selection, the game engine starts and thus prepares the necessary table for the gameplay.

The layout of the new game is shown in the adjacent figure. When the game is running the table will be filled with the values to form a Sudoku game and the timer is started to keep the track of time.

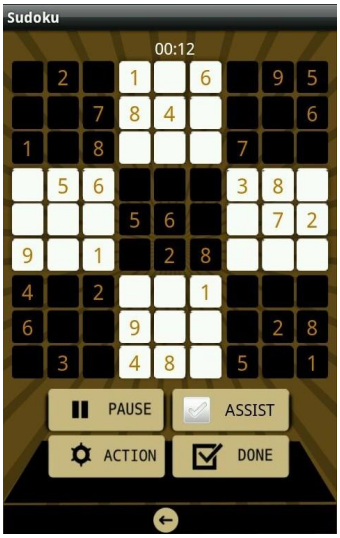


Fig. 5

The player also is entitled to the freedom of “Assist” option when is faced with difficulty while playing.

The different features of the New Game are:

1. Pause
2. Actions
    1. Restart Game
    2. New Game
    3. Check
    4. Quit
3. Done
4. Assist

The various actions' explanation are explained below

1. *Pause*

The pause button is used to pause the game while it is being played. On pressing the pause button the timer is stopped and a filler screen is shown which hides the content of the table till the user presses the resume button to resume the game.

Algorithm:

1. Stop the timer.
2. Display Filler Screen
3. Listen for Click on resume button
4. On- Resume, dismiss popup
5. Continue Timer.

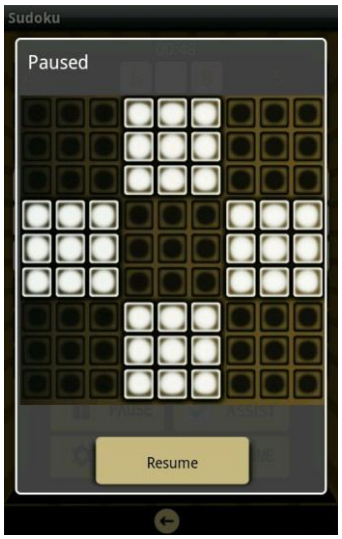


Fig. 5(a)

Flowchart

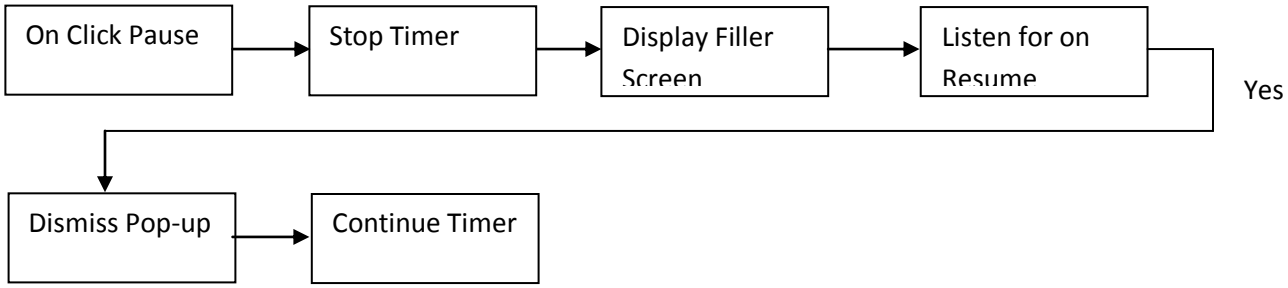


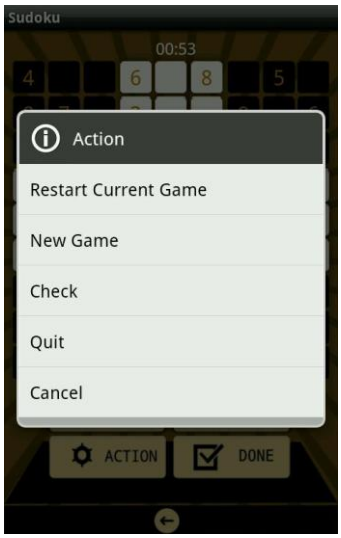
Fig. 5(b)

2. *Actions*

The Actions button provides the user with many more options that helps the user take control of the application. On clicking the button a context menu opens with various options. The options listed in the context menu are

1. Restart Current Game
2. Start New Game
3. Check
4. Quit

Fig. 5(c)



Algorithm: For Restart, Start and Quit

1. Listen for Clicks
2. Ask Confirmation
3. If yes, redirect to respective layout.

Flowchart

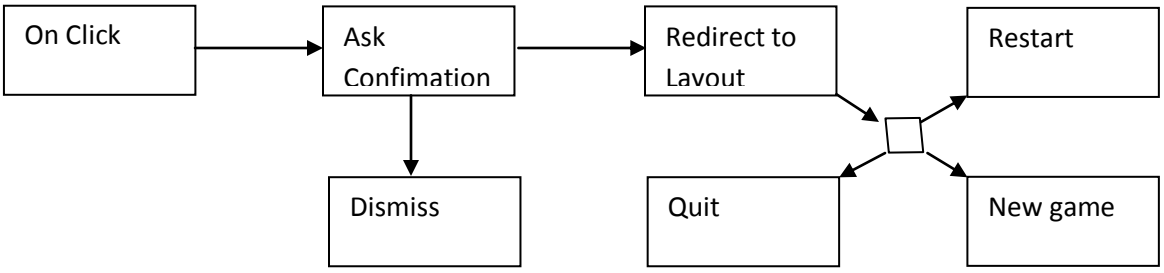


Fig. 5(d)

Algorithm: Check

1. Listen for Clicks
2. Check for repetition of a number in its respective row/column/cube.
3. If no mistake, display the right message
4. Else Display message with number of errors

Flowchart

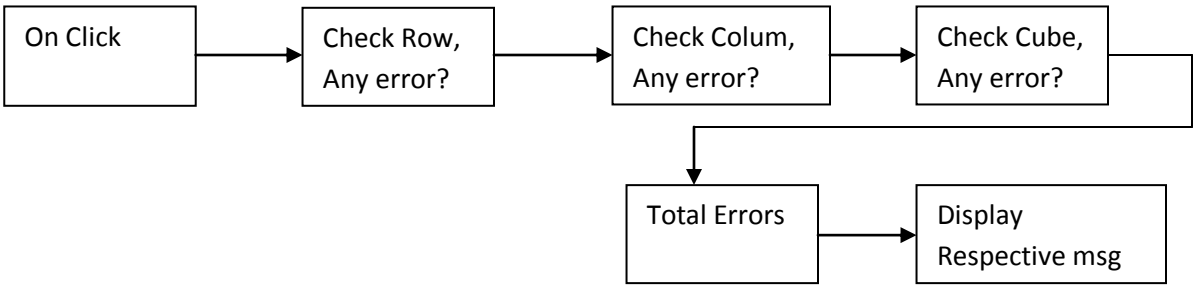


Fig. 5(e)

3. *Done*

"Done gives the user the freedom to submit the solution to check if the solution is correct or not. There are 3 possible outcomes on pressing the "done" button. They are :

1. "The table is not complete" – When there are unfilled cells.
2. "The solution is wrong" - When the solution is wrong.

   The above 2 notifications are shown as pop-up.
3. On completing the table successfully and Correctly, the user gets a "trophy" also Indicating the time taken.

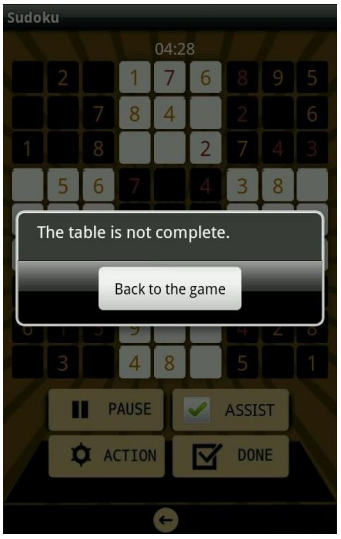


Fig. 5(f)

Algorithm

1. Listen for clicks
2. Call the checking function which check the possibility of appearance of a number in a cell.
3. Display appropriate message

Flowchart

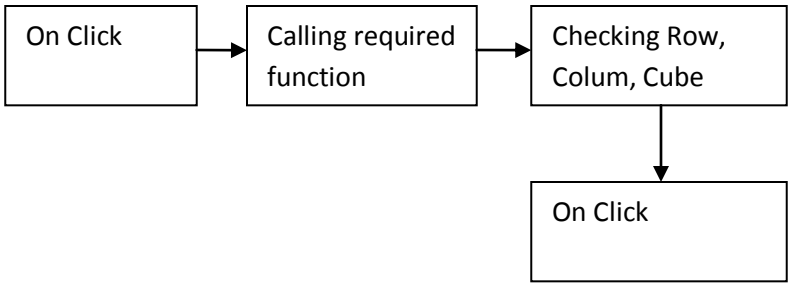
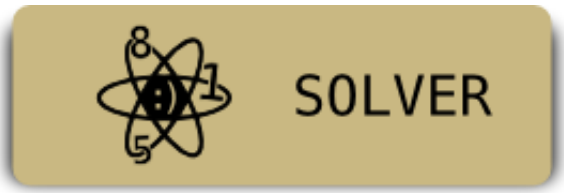


Fig. 5(g)

Fig. 5(h)



The app gives the user an unique feature that enables the player to get the solution of any Sudoku table by entering the values of the corresponding cell.

The user once after selecting the solver option is shown an empty table and the user is free to enter any values into the table. The smart entry feature ensures that only the values accepted to a particular cell which is calculated on a dynamic basis is only taken as input so as to completely reduce the error that can occur during input.

The page layout of the solver is shown alongside
The user is now free to enter any value in any cell.
clicking on any cell would then display the pop up
of keyboard. The numbers are thus available for entry.

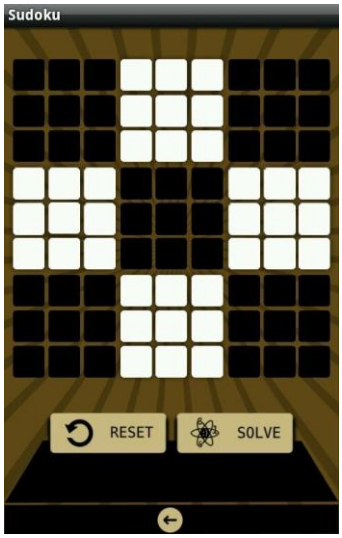
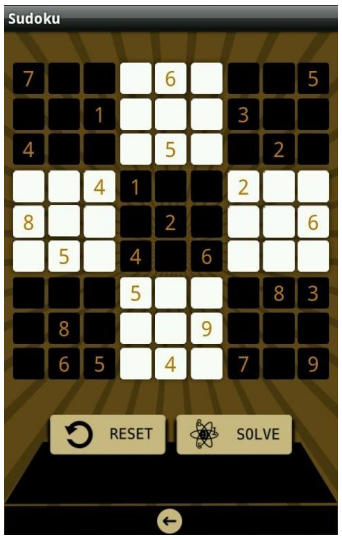


Fig. 6



The values entered by the user

Fig. 6(a)

The solution to the Sudoku table displayed after solving.

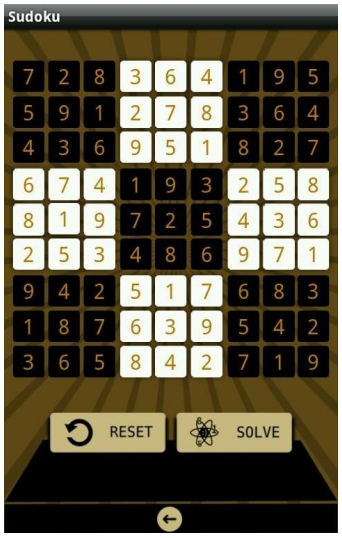


Fig. 6(b)

Algorithm

1. Take input values from user.
2. Validate the user input (with help of algorithm of smart assist)
3. Pass data to the function for finding the solution.
   (The algorithm for solving is by Brute-force method using recursion and back tracking strategy).
4. Display the final table after solving.

Flowchart

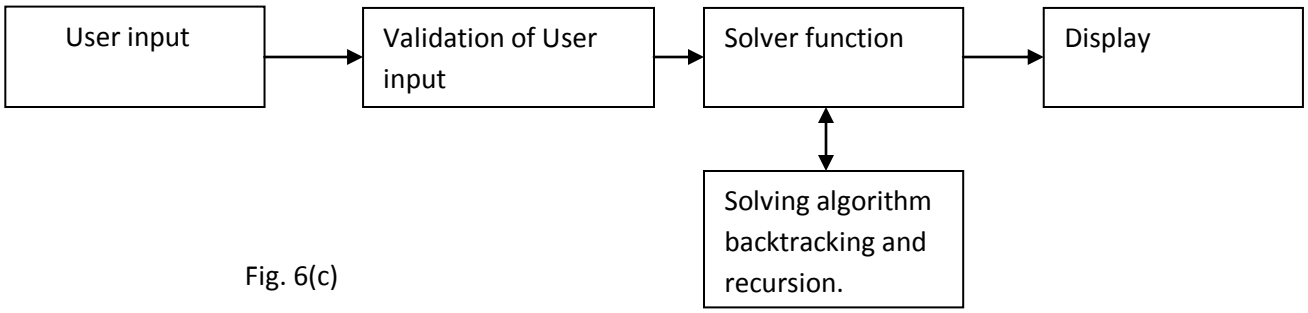


Fig. 6(c)

The solver also consists of options of “Reset” and “Solve”

1. Reset – Enables the user to clear the table for obtaining solution to another Sudoku table.
2. Solve – Enables the user to obtain the solution of the Sudoku table entered. Solve button functions as a signal/cue to the application to initiate solving procedure and display result.

On clicking on the Credits button, the control now shift to the credits layout page which displays the details of the key people involved in this project. The details are widely classified under *Code* and *Design*. On clicking the appropriate names a pop-up is created to provide provision of contact to any of the people involved.

The Layout of the “Credits” page is shown alongside. The names of the people involved are displayed as a ‘button’ at the UI level.

Thus enabling the properties of ‘button’ field of android, the user is now free to click on the names and thus it displays a ‘pop-up’ with the details. The ‘pop-up’ thus displayed can be dismissed by clicking on the ‘pop-up’.

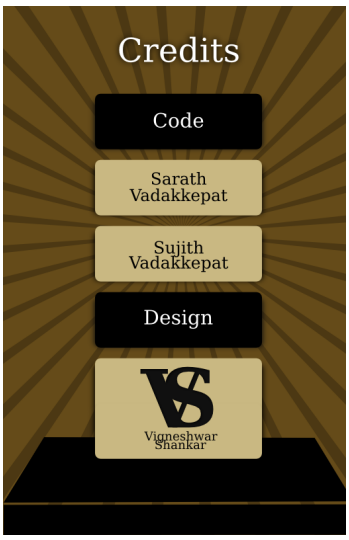


Fig. 7

Fig. 7(a)

Fig. 7(b)

Fig. 7(c)

The last option listed on the menu is the “Exit” option. This option is used to close all instances of the app and kill objects used by the program and free the memory and close the application.

This ensures that the application does not run any daemon process and all the threads and processes are killed and no memory is held. This ends the life cycle of the application.

APPENDIX

CODES

Codes

1. Main Activity

```
package com.sarath.sv17;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

public class Sudoku_SV17Activity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                final Intent mainIntent = new Intent(Sudoku_SV17Activity.this, Bg2.class);
                Sudoku_SV17Activity.this.startActivity(mainIntent);
                Sudoku_SV17Activity.this.finish();
               }
        }, 5000);
    }
}
```

2. Menu

```
package com.sarath.sv17;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class Bg2 extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
     {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.bgmenu);
        Button solver=null;
        Button new_game=null;
        Button exit=null;
        Button credits=null;
        solver=(Button)findViewById(R.id.button2);
        solver.setOnClickListener(new View.OnClickListener()
           {
              public void onClick(View view)
              {    Intent myIntent = new Intent(Bg2.this, Sudoku_solver.class);
                   startActivity(myIntent);
              }
           });
        new_game=(Button)findViewById(R.id.button1);
        new_game.setOnClickListener(new View.OnClickListener()
           {
               public void onClick(View view)
               {    Intent myIntent = new Intent(Bg2.this, GameEngine.class);
                startActivity(myIntent);
               }
             });
        exit=(Button)findViewById(R.id.button4);
        exit.setOnClickListener(new View.OnClickListener()
          {
            public void onClick(View view)
             {        System.exit(0);
              }
          });
        credits=(Button)findViewById(R.id.button3);
        credits.setOnClickListener(new View.OnClickListener()
           {
            public void onClick(View view)
               {    Intent myIntent = new Intent(Bg2.this, Credit.class);
                startActivity(myIntent);
                }
            });
        }
}
```

3. Game Engine

```
package com.sarath.sv17;

import android.app.Activity;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import java.util.Random;
import java.io.*;

public class GameEngine extends Activity
{
     String x;
     int i,j;
     int array[][]=new int[9][9];
     int ctr=0;

     @Override
    public void onCreate(Bundle savedInstanceState)
      {
           super.onCreate(savedInstanceState);
           toughness();
       }

    public void randomise()
      {
        ProgressDialog dialog = ProgressDialog.show(GameEngine.this, "", "Loading.",
true);
                    dialog.show();
                   int n1=random_func(4);
                    if(n1==1) random_level1();
                    else if(n1==2)
                    {   random_level1();
                        random_level2(); }
                    else if(n1==3)
                    {   random_level1();
                        random_level2();
                        random_level3(); }
                    else if(n1==4)
                    {   random_level1();
                        random_level2();
                        random_level3();
                        random_level4();  }
                    get_string();
                Intent intent =new Intent(GameEngine.this, Gameplay.class);
                intent.putExtra("question", x);
                startActivity(intent);
                finish();
          }
```

```
        public void toughness()
        {  final Context mContext = this;
           final Dialog dialog = new Dialog(mContext);
            dialog.setContentView(R.layout.toughness);
           dialog.setTitle("Level");
           dialog.show();
           Button Easy=(Button) dialog.findViewById(R.id.buttoneasy);
           Button Medium=(Button) dialog.findViewById(R.id.buttonmedium);
           Button Hard=(Button) dialog.findViewById(R.id.buttonhard);
           Easy.setOnClickListener(new View.OnClickListener()
            {    @Override
              public void onClick(View view)
              {   GetEasy();
                  dialog.dismiss();
                  randomise();
                  }
            });
           Medium.setOnClickListener(new View.OnClickListener()
            {   @Override
              public void onClick(View view)
              {   GetMedium();
                  randomise();
                  dialog.dismiss();
              }
            });
           Hard.setOnClickListener(new View.OnClickListener()
           {    @Override
              public void onClick(View view)
              {    GetHard();
                 randomise();
                 dialog.dismiss();
               }
           });
           return;
        }
        public void GetEasy()
        {  int easy1=random_func(80);
           int counter=0;
           try {
                  counter=0;
                  String str="";
                  InputStream is =
this.getResources().openRawResource(R.drawable.data_easy);
                  BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                  while ((str = reader.readLine()) != null)
                  {
                     counter++;
                     if(counter==easy1)
                         {      x=str;
                                break;
                         }
                  }
            }
            catch (IOException e) {  }
```

```
            return;
            }
            public void GetMedium()
            {  int medium1=random_func(80);
               int counter=0;
                  try {
                      counter=0;
                      String str="";
                      InputStream is =
this.getResources().openRawResource(R.drawable.data_medium);
                      BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                      while ((str = reader.readLine()) != null)
                      {  counter++;
                         if(counter==medium1)
                             {x=str;
                                    break;
                             }
                      }
                  }
                 catch (IOException e) {  }
            return;
            }
            public void GetHard()
            {  int hard1=random_func(80);
               int counter=0;
                try {
                      counter=0;
                      String str="";
                      InputStream is =
this.getResources().openRawResource(R.drawable.data_hard);
                      BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                      while ((str = reader.readLine()) != null)
                      {  counter++;
                         if(counter==hard1)
                             {      x=str;
                                    break;
                             }
                      }
                  }
                 catch (IOException e) {  }
           return;
            }
            public void random_level1()
               {   int n2=random_func(4);
                   if(n2==1) north();
                   else if(n2==2) south();
                   else if(n2==3) east();
                   else if(n2==4)  west();
                   return;
               }
               public void random_level2()
               {       int n2=random_func(3);
                       if(n2==1) mirror_vertical();
                       else if(n2==2) mirror_horizontal();
                       else if(n2==3) {}
```

```
     return;
}
public void random_level3()
{     int n2=random_func(36);
       switch(n2)
       {  case 1: break;
          case 2: shift_v(1,3,2);
                     break;
          case 3: shift_v(2,1,3);
                     break;
          case 4: shift_v(2,3,1);
                     break;
          case 5: shift_v(3,1,2);
                     break;
          case 6: shift_v(2,3,1);
                     break;
          case 7: shift_h(1,3,2);
              break;
          case 8: shift_h(1,3,2);
               shift_v(1,3,2);
              break;
           case 9: shift_h(1,3,2);
               shift_v(2,1,3);
              break;
          case 10: shift_h(1,3,2);
               shift_v(2,3,1);
              break;
          case 11: shift_h(1,3,2);
              shift_v(3,1,2);
              break;
          case 12: shift_h(1,3,2);
              shift_v(3,2,1);
              break;
          case 13: shift_h(2,1,3);
              break;
          case 14: shift_h(2,1,3);
              shift_v(1,3,2);
                     break;
          case 15: shift_h(2,1,3);
              shift_v(2,1,3);
                     break;
          case 16: shift_h(2,1,3);
              shift_v(2,3,1);
                     break;
          case 17: shift_h(2,1,3);
              shift_v(3,1,2);
                     break;
          case 18: shift_h(2,1,3);
              shift_v(3,2,1);
                     break;
          case 19: shift_h(2,3,1);
              break;
          case 20: shift_h(2,3,1);
              shift_v(1,3,2);
              break;
```

```
            case 21: shift_h(2,3,1);
                shift_v(2,1,3);
                break;
            case 22: shift_h(2,3,1);
                shift_v(2,3,1);
                break;
            case 23: shift_h(2,3,1);
                shift_v(3,1,2);
                break;
            case 24: shift_h(2,3,1);
                shift_v(3,2,1);
                 break;
            case 25: shift_h(3,1,2);
                break;
            case 26: shift_h(3,1,2);
                shift_v(1,3,2);
                break;
            case 27: shift_h(3,1,2);
                shift_v(2,1,3);
                break;
            case 28: shift_h(3,1,2);
                shift_v(2,3,1);
                break;
            case 29: shift_h(3,1,2);
                shift_v(3,1,2);
                break;
            case 30: shift_h(3,1,2);
                shift_v(3,2,1);
                break;
            case 31: shift_h(3,2,1);
                break;
            case 32: shift_h(3,2,1);
                shift_v(1,3,2);
                break;
            case 33: shift_h(3,2,1);
                shift_v(2,1,3);
                break;
            case 34: shift_h(3,2,1);
                shift_v(2,3,1);
                break;
            case 35: shift_h(3,2,1);
                shift_v(3,1,2);
                break;
            case 36: shift_h(3,2,1);
                shift_v(3,2,1);
                break;
        }
        return;
}
public void shift_h(int a, int b, int c)
{  east();
   shift_v(a,b,c);
    west();
  return;
}
```

```
public void shift_hm(int a, int b, int c)
{  east();
       shift_vm(a,b,c);
  west();
  return;
}
public void random_level4()
{
       int n2=random_func(3);
              if(n2==1) mirror_vertical();
              else if(n2==2) mirror_horizontal();
              else if(n2==3)
              {
                     n2=random_func(2);
                     if(n2==2) supercharged();
              }

       return;
}
public void supercharged()
{  int n2=random_func(36);
       switch(n2)
       {
          case 1: break;
         case 2: shift_vm(1,3,2);
                     break;
          case 3: shift_vm(2,1,3);
                     break;
           case 4: shift_vm(2,3,1);
                     break;
           case 5: shift_vm(3,1,2);
                     break;
           case 6: shift_vm(2,3,1);
                     break;
           case 7: shift_hm(1,3,2);
               break;
           case 8: shift_hm(1,3,2);
                shift_vm(1,3,2);
               break;
           case 9: shift_hm(1,3,2);
                shift_vm(2,1,3);
               break;
           case 10: shift_hm(1,3,2);
                shift_vm(2,3,1);
               break;
           case 11: shift_hm(1,3,2);
               shift_vm(3,1,2);
               break;
           case 12: shift_hm(1,3,2);
               shift_vm(3,2,1);
               break;
           case 13: shift_hm(2,1,3);
               break;
           case 14: shift_hm(2,1,3);
               shift_vm(1,3,2);
```

```
            break;
case 15: shift_hm(2,1,3);
    shift_vm(2,1,3);
            break;
case 16: shift_hm(2,1,3);
    shift_vm(2,3,1);
            break;
case 17: shift_hm(2,1,3);
    shift_vm(3,1,2);
            break;
case 18: shift_hm(2,1,3);
    shift_vm(3,2,1);
            break;
case 19: shift_hm(2,3,1);
    break;
case 20: shift_hm(2,3,1);
    shift_vm(1,3,2);
    break;
case 21: shift_hm(2,3,1);
    shift_vm(2,1,3);
    break;
case 22: shift_hm(2,3,1);
    shift_vm(2,3,1);
    break;
case 23: shift_hm(2,3,1);
    shift_vm(3,1,2);
    break;
case 24: shift_hm(2,3,1);
    shift_vm(3,2,1);
     break;
case 25: shift_hm(3,1,2);
    break;
case 26: shift_hm(3,1,2);
    shift_vm(1,3,2);
    break;
case 27: shift_hm(3,1,2);
    shift_vm(2,1,3);
    break;
case 28: shift_hm(3,1,2);
    shift_vm(2,3,1);
    break;
case 29: shift_hm(3,1,2);
    shift_vm(3,1,2);
    break;
case 30: shift_hm(3,1,2);
    shift_vm(3,2,1);
    break;
case 31: shift_hm(3,2,1);
    break;
case 32: shift_hm(3,2,1);
    shift_vm(1,3,2);
    break;
case 33: shift_hm(3,2,1);
    shift_vm(2,1,3);
    break;
```

```
            case 34: shift_hm(3,2,1);
                shift_vm(2,3,1);
                break;
            case 35: shift_hm(3,2,1);
                shift_vm(3,1,2);
                break;
            case 36: shift_hm(3,2,1);
                shift_vm(3,2,1);
                break;
        }
        return;
}
public int random_func(int n)
{       Random rand=new Random();
        int num=rand.nextInt(n);
        return num+1;
}
public void north()
{    ctr=0;
        for(i=0;i<9;i++)
            for(j=0;j<9;j++)
               array[i][j]=Integer.parseInt(String.valueOf(x.charAt(ctr++)));
    get_string();
    return;
}
public void south()
{    east();
    east();
    get_string();
    return;
}
public void west()
{       ctr=0;
        for(i=0;i<9;i++)
            for(j=8;j>=0;j--)
               array[j][i]=Integer.parseInt(String.valueOf(x.charAt(ctr++)));
        get_string();
        return;          }
public void east()
{        ctr=0;
        for(i=8;i>=0;i--)
          for(j=0;j<9;j++)
            array[j][i]=Integer.parseInt(String.valueOf(x.charAt(ctr++)));
    get_string();
    return;
 }
public void get_string()
{    x="";
        for(i=0;i<9;i++)
            for(j=0;j<9;j++)
               x=x+String.valueOf(array[i][j]);

         return;
}
public void mirror_vertical()
```

```
    {      ctr=0;
           for(i=0;i<9;i++)
              for(j=8;j>=0;j--)
                  array[i][j]=Integer.parseInt(String.valueOf(x.charAt(ctr++)));

        get_string();
        return;
    }
 public void mirror_horizontal()
    {      ctr=0;
           for(i=8;i>=0;i--)
              for(j=0;j<9;j++)
                  array[i][j]=Integer.parseInt(String.valueOf(x.charAt(ctr++)));

        get_string();
        return;
     }
public void shift_v(int a,int b,int c)
   {
    String str[]=new String[3];
    str[0]=x.substring(0,27);
    str[1]=x.substring(27,54);
    str[2]=x.substring(54,81);
    for(i=0;i<3;i++)
    {
           for(int j=0;j<9;j++)
              array[i][j]=str[a-1].charAt(i*9+j)-48;
           for(int j=0;j<9;j++)
              array[i+3][j]=str[b-1].charAt(i*9+j)-48;
           for(int j=0;j<9;j++)
              array[i+6][j]=str[c-1].charAt(i*9+j)-48;
    }
    get_string();
    return;
   }
public void shift_vm(int a,int b,int c)
   { String str[]=new String[3];
    str[0]=x.substring(27,36);
    str[1]=x.substring(36,45);
    str[2]=x.substring(45,54);

           for(int j=0;j<9;j++)
              array[3][j]=str[a-1].charAt(j)-48;
           for(int j=0;j<9;j++)
              array[4][j]=str[b-1].charAt(j)-48;
           for(int j=0;j<9;j++)
              array[5][j]=str[c-1].charAt(j)-48;
    get_string();
    return;
   }
```

4. Game Engine – Alternate

```
package com.sarath.sv17;

import android.app.Activity;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import java.util.Random;
import java.io.*;

public class GameEngine_alternate extends Activity{

        String x="";
        String x1="";
    @Override
    public void onCreate(Bundle savedInstanceState)
          {
           super.onCreate(savedInstanceState);
           toughness();
         }

    public void randomise()
          {
              ProgressDialog dialog = ProgressDialog.show(GameEngine_alternate.this, "",
"Loading.", true);
              dialog.show();
              int n1=random_func(4);
                   if(n1==1) random_level1();
                   else if(n1==2) random_level2();
                   else if(n1==3) random_level3();
                   else if(n1==4) random_level4();
               Intent intent =new Intent(GameEngine_alternate.this, Gameplay.class);
               intent.putExtra("question", x);
               startActivity(intent);
               finish();
          }
          public void toughness()
          {
             final Context mContext = this;
             final Dialog dialog = new Dialog(mContext);
                   dialog.setContentView(R.layout.toughness);
             dialog.setTitle("Level");
             dialog.show();
             Button Easy=(Button) dialog.findViewById(R.id.buttoneasy);
             Button Medium=(Button) dialog.findViewById(R.id.buttonmedium);
             Button Hard=(Button) dialog.findViewById(R.id.buttonhard);
             Easy.setOnClickListener(new View.OnClickListener()
          {
               @Override
```

```
            public void onClick(View view)
            {
                    GetEasy();
                dialog.dismiss();
                randomise();
                }
        });
           Medium.setOnClickListener(new View.OnClickListener()
        {
             @Override
            public void onClick(View view)
            {
                    GetMedium();
                randomise();
                dialog.dismiss();
            }
        });
           Hard.setOnClickListener(new View.OnClickListener()
        {
             @Override
            public void onClick(View view)
            {
                   GetHard();
               randomise();
               dialog.dismiss();
             }
        });
           return;
        }
       public void GetEasy()
        {  int easy1=random_func(80);
           int counter=0;
           try {
                  counter=0;
                  String str="";
                  InputStream is =
this.getResources().openRawResource(R.drawable.data_easy);
                  BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                  while ((str = reader.readLine()) != null)
                  {  counter++;
                     if(counter==easy1)
                         {   x=str;
                             break;
                         }
                  }
             }
            catch (IOException e) {  }
       return;
        }
        public void GetMedium()
        {  int medium1=random_func(80);
           int counter=0;

            try {
                  counter=0;
```

```
                    String str="";
                    InputStream is =
this.getResources().openRawResource(R.drawable.data_medium);
                    BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                    while ((str = reader.readLine()) != null)
                    {  counter++;
                       if(counter==medium1)
                           {      x=str;
                                  break;
                           }
                    }
                 }
                catch (IOException e) {  }
            return;
            }
            public void GetHard()
            {  int hard1=random_func(80);
               int counter=0;

                try {
                     counter=0;
                     String str="";
                     InputStream is =
this.getResources().openRawResource(R.drawable.data_hard);
                     BufferedReader reader = new BufferedReader(new InputStreamReader(is));
                     while ((str = reader.readLine()) != null)
                     {  counter++;
                        if(counter==hard1)
                           {      x=str;
                                  break;
                           }
                     }
                 }
                catch (IOException e) {  }
            return;
            }
            public void random_level1()
               {
                   int n2=random_func(4);
                   if(n2==2) smart(8,1,0,8,1,0,1,0);
                      else if(n2==3) smart(0,-1,8,8,1,0,2,0);
                      else if(n2==4) smart(8,1,0,8,1,0,1,0);
                      return;
               }
            public void random_level2()
               {      random_level1();
                      int n2=random_func(3);
                   if(n2==1) smart(0,-1,8,8,1,0,1,0);
                   else if(n2==2) smart(8,1,0,0,-1,8,1,0);
                      return;
               }
               public void random_level3()
               {
                   random_level2();
                      int n2=random_func(36)-1;
```

```
                int a1=(int)(n2/6);
                int a2=n2-(a1*6);
                int array[]={1,2,3,1,3,2,2,1,3,2,3,1,3,1,2,3,2,1};
                shift_h(array[a1*3],array[(a1*3)+1],array[(a1*3)+2]);
                shift_v(array[a2*3],array[(a2*3)+1],array[(a2*3)+2]);
                return;
            }
            public void random_level4()
            {       random_level3();
                random_level2();
                int n2=random_func(2);
                if(n2==2)
                 {       n2=random_func(2);
                        if(n2==2) supercharged();
                 }
                return;
             }
             public int random_func(int n)
            {
                Random rand=new Random();
               return (rand.nextInt(n)+1);
            }
            public void smart(int a,int b,int c,int a1,int b1,int c1,int code,int e)
             {
            if(e==0) x="";
            for(int i=a;(-i*b)<=c;i=i-b)
            {  for(int j=a1;(-j*b1)<=c1;j=j-b1)
                {  if(code==1) x=x+x1.charAt((i*9)+j);
                        if(code==2)  x=x+x1.charAt((j*9)+i);
                            }
                               }
            if(e==0) x1=x;
             }
            public void shift_v(int xx, int yy, int zz)
             {
              x="";
                 smart(((xx-1)*3),-1,((xx-1)*3)+2,0,-1,8,1,1);
                 smart(((yy-1)*3),-1,((yy-1)*3)+2,0,-1,8,1,1);
               smart(((zz-1)*3),-1,((zz-1)*3)+2,0,-1,8,1,1);
               x1=x;
              }
             public void shift_h(int xx, int yy, int zz)
             {
            x="";
                int num=0;
                while(num<=8)
                {  x=x+x1.substring((num*9)+((xx-1)*3),((num*9)+((xx-
1)*3))+3)+x1.substring((num*9)+((yy-1)*3),((num*9)+((yy-1)*3))+3);
                   x=x+x1.substring((num*9)+((zz-1)*3),((num*9)+((zz-1)*3))+3);
                   num++;
                   }
             x1=x;
              }
             public void shift_vm(int xx,int yy,int zz)
            {
```

```
                x="";
                x=x+x1.substring(0,27);
              smart(xx+2,-1,xx+2,0,-1,8,1,1);
              smart(yy+2,-1,yy+2,0,-1,8,1,1);
              smart(zz+2,-1,zz+2,0,-1,8,1,1);
              x=x+x1.substring(54,81);
              x1=x;
           }
          public void shift_hm(int xx, int yy, int zz)
           {
             x="";
                int num=0;
             while(num<=8)
                {
                  x=x+x1.substring((num*9),(num*9)+3);

x=x+x1.charAt((num*9)+xx+2)+x1.charAt((num*9)+yy+2)+x1.charAt((num*9)+zz+2);
                  x=x+x1.substring((num*9)+6,((num*9)+6)+3);
                    num++;
                }
           x1=x;
            }
           public void supercharged()
             {
                    int n2=random_func(36)-1;
                    int a1=(int)(n2/6);
                    int a2=n2-(a1*6);
                    int array[]={1,2,3,1,3,2,2,1,3,2,3,1,3,1,2,3,2,1};
                    shift_hm(array[a1*3],array[(a1*3)+1],array[(a1*3)+2]);
                    shift_vm(array[a2*3],array[(a2*3)+2]);
                    return;
             }

       }
```

5. Game Play

```
package com.sarath.sv17;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AbsoluteLayout;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.CheckBox;
import android.widget.Chronometer;
import android.widget.Toast;
import android.os.SystemClock;

public class Gameplay extends Activity {
    /** Called when the activity is first created. */

    Chronometer mChronometer;

       int sudoku_array_game[][]=new int[9][9];
       int check_array[][]=new int[9][9];
       Button numarray_button[]=new Button[10];
       int array_button_dialog[]=new int[10];
       TextView display_textview_array[]=new TextView[81];
       EditText display_EditText_array[]=new EditText[81];
       String x;
       View v2;
       int to_return=0, id1, toggle=0, flag_done=0;
       int cell_id;
       long elapsedMillis=0;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.new_play);

        EditText num00=(EditText) findViewById(R.id.num00);
        cell_id=num00.getId();
        mChronometer = (Chronometer) findViewById(R.id.chronometer1);
        x=getIntent().getExtras().getString("question");
        start();
```

```
    CheckBox chkIos = (CheckBox) findViewById(R.id.checkBox1);
    chkIos.setOnClickListener(new OnClickListener()
    {      @Override
     public void onClick(View v)
            {
         if (((CheckBox) v).isChecked())
          toggle=1;
          else
                toggle=0;
            }
  });

}
 public int check_between()
 {
   int flag=0;
   capturescreen();
         for(int i=0;i<9;i++)
            for(int j=0;j<9;j++)
                  if(check_array[i][j]==0&&sudoku_array_game[i][j]!=0)
                       if(!(val(sudoku_array_game[i][j],i,j)))
                          flag++;

   return flag;
 }

 public boolean val(int num,int r,int c)
  {
         int ctr=0;
         for(int i=0;i<9;i++)
            if(sudoku_array_game[r][i]==num)
              ctr++;
         if(ctr!=1)
            return false;
         ctr=0;

         for(int i=0;i<9;i++)
           if(sudoku_array_game[i][c]==num)
              ctr++;
         if(ctr!=1)
            return false;
         ctr=0;

         r = (r / 3) * 3 ;
     c = ( c/ 3) * 3 ;
     for( int i = 0;i<3;i++ )
       for( int j = 0;j<3;j++ )
        if(sudoku_array_game[r+i][c+j]==num)
          ctr++;
     if(ctr!=1)
            return false;

     return true ;
  }
  public void done()
```

```
      {
       flag_done=0;
            for(int i=0;i<9;i++)
                  for(int j=0;j<9;j++)
                        if(sudoku_array_game[i][j]==0)
                        {
                              flag_done=1;
                            return;
                        }

            for(int i=0;i<9;i++)
                  for(int j=0;j<9;j++)
                        if(check_array[i][j]==0)
                           if(!(val(sudoku_array_game[i][j],i,j)))
                             flag_done=2;

        return;
      }

    public void popup_done()
    {
       stop_timer();
          final Context mContext = this;
            final Dialog dialog = new Dialog(mContext);
                  dialog.setContentView(R.layout.highscore);
                  dialog.setTitle("                Congratulations");
            dialog.show();
            RelativeLayout rl1=(RelativeLayout) dialog.findViewById(R.id.done);
            rl1.setOnClickListener(new View.OnClickListener()
                {
                    public void onClick(View view)
                    {
                                dialog.dismiss();
                                back_to_main();
                     }
                  });
            int min=(int)(elapsedMillis/60000);
            int sec=(int)((elapsedMillis/1000)-min*60);
            TextView pop_do=(TextView) dialog.findViewById(R.id.textView1);
            pop_do.setText("You have solved the puzzle in \n"+min+" Minute(s) "+sec+"
Second(s).");

            reset_timer();
            return;
      }

    public void back_to_main()
    {
       AlertDialog.Builder builder = new AlertDialog.Builder(this);
              builder.setMessage("Next is what?")
             .setCancelable(false)
             .setPositiveButton("Main menu", new DialogInterface.OnClickListener() {
                 public void onClick(DialogInterface dialog, int id) {
                     finish();
                 }
```

```
            })
            .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                     dialog.cancel();
                }
            })
            .setNeutralButton("New game", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                     Intent myIntent = new Intent(Gameplay.this, GameEngine.class);
                    startActivity(myIntent);
                    finish();
                }
            });
              AlertDialog alert = builder.create();
              alert.show();

              return;
     }
    public void start()
    {
       start_timer();
       put_table();
        setup_display_textview();
        setup_edittext_listener();
        display();

        Button button1=(Button)findViewById(R.id.button_solve);
        button1.setOnClickListener(new OnClickListener()
             {
                @Override
                public void onClick(View v)
                {
                  registerForContextMenu(v);
               openContextMenu(v);
               unregisterForContextMenu(v);
             }});

        Button button2=(Button)findViewById(R.id.reset);
             button2.setOnClickListener(new OnClickListener()
             {
               @Override
               public void onClick(View v)
               {
                     capturescreen();
                     done();
                     popup_alert();

            }});

             Button pause=(Button)findViewById(R.id.pause);
             pause.setOnClickListener(new OnClickListener()
             {
                @Override
                public void onClick(View v)
                {
```

```
                stop_timer();
              pause_pop();
               }});

        Button bback=(Button)findViewById(R.id.b_back);
        bback.setOnClickListener(new OnClickListener()
        {
           @Override
           public void onClick(View v)
           {
                  alerts(1);
                }});
 }

public void alerts(int code)
{
   final int cd=code;
   String show="";
   String positive="Yes";
   String negative="No";

   if(code==1)
          show="Back, are you sure?";
   if(code==2)
          show="Restart, are you sure?";
   if(code==3)
          show="New Game, are you sure?";
   if(code==4)
          show="Quit, are you sure?";

          AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setMessage(show)
           .setCancelable(false)
           .setPositiveButton(positive, new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int id)

               {
                   if(cd==1||cd==4) finish();
                   if(cd==2)
                   {
                      reset_timer();
                      start();
                   }
                   if(cd==3)
                   {
                        Intent myIntent = new Intent(Gameplay.this, GameEngine.class);
                    startActivity(myIntent);
                    finish();
                   }

               }
           })
           .setNegativeButton(negative, new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int id) {
                    dialog.cancel();
```

```
                }
            });
             AlertDialog alert = builder.create();
             alert.show();
             return;
    }

     public void start_timer()
     {
       int stoppedMilliseconds = 0;
        String chronoText = mChronometer.getText().toString();
        String array[] = chronoText.split(":");
        if (array.length == 2)
        {
          stoppedMilliseconds = Integer.parseInt(array[0]) * 60 * 1000
              + Integer.parseInt(array[1]) * 1000;
        }
        else if (array.length == 3)
        {
          stoppedMilliseconds = Integer.parseInt(array[0]) * 60 * 60 * 1000
              + Integer.parseInt(array[1]) * 60 * 1000
              + Integer.parseInt(array[2]) * 1000;
        }
        mChronometer.setBase(SystemClock.elapsedRealtime() - stoppedMilliseconds);
        mChronometer.start();
    }

    public void stop_timer()
    {
       mChronometer.stop();
        elapsedMillis = SystemClock.elapsedRealtime()- mChronometer.getBase();
    }

    public void reset_timer()
    {
       mChronometer.setBase(SystemClock.elapsedRealtime());
        mChronometer.stop();
    }

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,ContextMenuInfo menuInfo)
    {
      super.onCreateContextMenu(menu, v, menuInfo);
      menu.setHeaderTitle("Action");
      menu.add(0, v.getId(), 0, "Restart Current Game");
      menu.add(0, v.getId(), 0, "New Game");
      menu.add(0, v.getId(), 0, "Check");
      menu.add(0, v.getId(), 0, "Quit");
      menu.add(0, v.getId(), 0, "Cancel");
   }
    public void popup_alert()
    {
       AlertDialog.Builder builder = new AlertDialog.Builder(this);
      if(flag_done==1||flag_done==2)
      {
```

```
        if(flag_done==1)
        {
                 builder.setMessage("The table is not complete.")
                 .setCancelable(false)
                .setPositiveButton("Back to the game", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                         dialog.cancel();
                    }
                });
        }
               if(flag_done==2)
               {
                 builder.setMessage("Sorry, the solution is incorrect.")
                 .setCancelable(false)
                .setPositiveButton("Back to the game", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                         dialog.cancel();
                    }
                });
               }

          AlertDialog alert = builder.create();
          alert.show();
        }
        if(flag_done==0)
               {
                      popup_done();
                      return;
               }
    }

    @Override
    public boolean onContextItemSelected(MenuItem item)
    {

        if(item.getTitle()=="Restart Current Game")  alerts(2);
        else if(item.getTitle()=="New Game") alerts(3);
        else if(item.getTitle()=="Cancel")    return false;
        else if(item.getTitle()=="Quit") alerts(4);

        else if(item.getTitle()=="Check")
        {
             AlertDialog.Builder builder = new AlertDialog.Builder(this);
             int errors=check_between();
             if(errors==0)
             {
                 builder.setMessage("So far so good.")
                    .setCancelable(false)
                    .setPositiveButton("Back to the game", new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id) {
                             dialog.cancel();
                        }
                    });
               }
```

```
            if(errors>0)
            {
                            builder.setMessage("You have made "+errors+" error(s).")
                   .setCancelable(false)
                   .setPositiveButton("Back to the game", new
DialogInterface.OnClickListener() {
                       public void onClick(DialogInterface dialog, int id) {
                            dialog.cancel();
                       }

                   });
            }
             AlertDialog alert = builder.create();
            alert.show();
       }

       else
       {
         return false;
        }

    return true;
    }

    public void pause_pop()
    {
       final Context mContext = this;
       final Dialog dialog = new Dialog(mContext);
              dialog.setContentView(R.layout.poppause);
       dialog.setTitle("Paused");
       dialog.show();

       Button back=(Button) dialog.findViewById(R.id.resume);
       back.setOnClickListener(new View.OnClickListener()
       {   @Override
             public void onClick(View view)
             {
              dialog.dismiss();
              start_timer();
             }
       });

       return;
   }

    public void setup_edittext_listener()
    {
       final AbsoluteLayout a1=(AbsoluteLayout) findViewById(R.id.abs2);
        for(int i=0;i<81;i++)
        {
            v2 = a1.getChildAt(i);
            if(v2 instanceof EditText)
            {
                EditText a2=(EditText) v2;
                a2.setOnClickListener(new View.OnClickListener()
```

```
            {
                    @Override
                    public void onClick(View view) {
                    int ids=view.getId();
                    if(toggle==1)
                       check1(ids);
                    if(toggle==0)
                           for(int i1=0;i1<10;i1++)
                                   array_button_dialog[i1]=0;
                      numBoard(ids);
                }
             });
     }
   }
 }

public void dialog_function_display()
{
   final AbsoluteLayout a3=(AbsoluteLayout) findViewById(R.id.abs2);
   int diff=id1-cell_id;
    TextView temp=(TextView) a3.getChildAt(diff);
    temp.setTextColor(0xff993333);
    if(to_return!=0)
    temp.setText(String.valueOf(to_return));
    if(to_return==0)
    temp.setText(String.valueOf(""));
    capturescreen();

    return;
}

public void check1(int ids1)
{
   int i2=0;
    int xstart=0,ystart=0;
   display();
   int cellcode=ids1-cell_id;
   int xx1=(int)(cellcode/9);
   int xx2=cellcode-(9*xx1);
   for(int i1=0;i1<10;i1++)
          array_button_dialog[i1]=0;

    i2=xx1;
    for(int j2=0;j2<9;j2++)
       array_button_dialog[sudoku_array_game[i2][j2]]=1;

     i2=xx2;
    for(int j2=0;j2<9;j2++)
         array_button_dialog[sudoku_array_game[j2][i2]]=1;

    if(xx1<3) xstart=0;
    if(xx1>2&&xx1<6) xstart=3;
    if(xx1>5) xstart=6;
    if(xx2<3) ystart=0;
    if(xx2>2&&xx2<6) ystart=3;
```

```
        if(xx2>5) ystart=6;

        for(int k=xstart;k<xstart+3;k++)
             for(int k1=ystart;k1<ystart+3;k1++)
                   array_button_dialog[sudoku_array_game[k][k1]]=1;

       return;
   }

    public void capturescreen()
    {
       setup_display_edittext();
       int ctr=0;
       for(int i=0;i<9;i++)
                 for(int j=0;j<9;j++)

sudoku_array_game[i][j]=returnstr(display_EditText_array[ctr++].getText().toString());

     return;
    }

    public int returnstr(String input1)
    {
       if(input1.length()==0)
             return 0;
       else
             return Integer.parseInt(input1);
    }

    public void numBoard(int id2)
    {
       id1=id2;
       final Context mContext = this;
       final Dialog dialog = new Dialog(mContext);
       dialog.setContentView(R.layout.keyboard);
       dialog.setTitle("NumBoard");
       dialog.show();
       numarray_button[0]=(Button) dialog.findViewById(R.id.num0);
       numarray_button[1]=(Button) dialog.findViewById(R.id.num1);
       numarray_button[2]=(Button) dialog.findViewById(R.id.num2);
       numarray_button[3]=(Button) dialog.findViewById(R.id.num3);
       numarray_button[4]=(Button) dialog.findViewById(R.id.num4);
       numarray_button[5]=(Button) dialog.findViewById(R.id.num5);
       numarray_button[6]=(Button) dialog.findViewById(R.id.num6);
       numarray_button[7]=(Button) dialog.findViewById(R.id.num7);
       numarray_button[8]=(Button) dialog.findViewById(R.id.num8);
       numarray_button[9]=(Button) dialog.findViewById(R.id.num9);

       for(int h=1;h<10;h++)
          if(array_button_dialog[h]==1)
                   numarray_button[h].setEnabled(false);

       for(int h=1;h<10;h++)
             if(array_button_dialog[h]==0)
                   numarray_button[h].setBackgroundResource(R.drawable.b_assist);
```

```
numarray_button[1].setOnClickListener(new View.OnClickListener()
    {  @Override
         public void onClick(View view)
         {
             to_return=1;
             dialog_function_display();
             dialog.dismiss();
        }
    });

numarray_button[2].setOnClickListener(new View.OnClickListener()
{    @Override
      public void onClick(View view)
      {
       to_return=2;
       dialog_function_display();
       dialog.dismiss();
       }
});

numarray_button[3].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {
       to_return=3;
       dialog_function_display();
       dialog.dismiss();
       }
});

numarray_button[4].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=4;
       dialog_function_display();
       dialog.dismiss();
       }
});

numarray_button[5].setOnClickListener(new View.OnClickListener()
{  @Override
      public void onClick(View view)
      {       to_return=5;
       dialog_function_display();
       dialog.dismiss();
       }
});

numarray_button[6].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {
       to_return=6;
       dialog_function_display();
```

```
            dialog.dismiss();
            }
        });
        numarray_button[7].setOnClickListener(new View.OnClickListener()
        {    @Override
             public void onClick(View view)
             {       to_return=7;
              dialog_function_display();
              dialog.dismiss();
              }
        });
        numarray_button[8].setOnClickListener(new View.OnClickListener()
        {    @Override
             public void onClick(View view)
             {       to_return=8;
              dialog_function_display();
              dialog.dismiss();
              }
        });
        numarray_button[9].setOnClickListener(new View.OnClickListener()
        {   @Override
             public void onClick(View view)
             {        to_return=9;
               dialog_function_display();
              dialog.dismiss();
              }
        });
        numarray_button[0].setBackgroundResource(R.drawable.b_eraser);
        numarray_button[0].setOnClickListener(new View.OnClickListener()
        {   @Override
             public void onClick(View view)
             {        to_return=0;
               dialog_function_display();
              dialog.dismiss();
              }
        });
         Button back=(Button) dialog.findViewById(R.id.numback);
        back.setEnabled(false);
        Button back1=(Button) dialog.findViewById(R.id.numback1);
        back1.setBackgroundResource(R.drawable.b_x);
        back1.setOnClickListener(new View.OnClickListener()
        {   @Override
             public void onClick(View view)
             {
              dialog.dismiss();
              }
        });
        return;
     }
    public void setup_display_edittext()
      {
           AbsoluteLayout llMain = (AbsoluteLayout)findViewById(R.id.abs2);
           for(int i=0;i<81;i++)
         {
            View v1 = llMain.getChildAt(i);
```

```
            if(v1 instanceof EditText)
               display_EditText_array[i]=(EditText) v1;
            }
            return;
     }
    public void put_table()
     {
        for(int i=0;i<9;i++)
               for(int j=0;j<9;j++)
                     check_array[i][j]=0;

        int ctr1=0;
        for(int i=0;i<9;i++)
        {
               for(int j=0;j<9;j++)
               {

        sudoku_array_game[i][j]=Integer.parseInt(String.valueOf(x.charAt(ctr1++)));
                     if(sudoku_array_game[i][j]!=0)
                            check_array[i][j]=1;
               }
        }
        return;
     }
    public void setup_display_textview()
     {
        AbsoluteLayout llMain = (AbsoluteLayout)findViewById(R.id.abs2);
         for(int i=0;i<81;i++)
            {
                  View v = llMain.getChildAt(i);
                  if(v instanceof TextView)
              {
                     ((TextView)v).setTextColor(0xffbe7b00);
                   display_textview_array[i]=(TextView) v;
              }
            }
    }
    public void display()
     {  int ctr=0;
         for(int i=0;i<9;i++)
               for(int j=0;j<9;j++)
               {   String string_dis=display_String(sudoku_array_game[i][j]);
                   display_textview_array[ctr++].setText(string_dis);
               }
         return;
     }
     public String display_String(int num)
     {  String g="";
        if(num==0)
        return g;
        else
        return String.valueOf(num);
     }

}
```

6. Solver

```
package com.sarath.sv17;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AbsoluteLayout;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import android.content.Context;
import android.content.DialogInterface;
import android.app.Dialog;

public class Sudoku_solver extends Activity{

       int sudoku_array[][]=new int[9][9];
       TextView display_textview_array[]=new TextView[81];
       EditText display_EditText_array[]=new EditText[81];
       EditText input_string;

       int cell_id;
       View v2;
       int to_return=0, id1;
       Button numarray_button[]=new Button[10];
       int array_button_dialog[]=new int[10];


    @Override
    public void onCreate(Bundle savedInstanceState)
    {   super.onCreate(savedInstanceState);
        setContentView(R.layout.sudoku_solver);
        setup_display_textview();
        EditText num00=(EditText) findViewById(R.id.num00);
        cell_id=num00.getId();
        start();
     }

    public void setup_display_textview()
    {  AbsoluteLayout llMain = (AbsoluteLayout)findViewById(R.id.abs2);
        for(int i=0;i<81;i++)
           {   View v = llMain.getChildAt(i);
               if(v instanceof TextView)
               {    ((TextView)v).setTextColor(0xffbe7b00);
                  display_textview_array[i]=(TextView) v;  }}
       return;
    }

    public void setup_display_edittext()
```

```
      {  AbsoluteLayout llMain = (AbsoluteLayout)findViewById(R.id.abs2);
       for(int i=0;i<81;i++)
        {  View v1 = llMain.getChildAt(i);
           if(v1 instanceof EditText)
              display_EditText_array[i]=(EditText) v1;    }
       return;
      }
    public void capturescreen()
    {  setup_display_edittext();
       int ctr=0;
       for(int i=0;i<9;i++)
                 for(int j=0;j<9;j++)

sudoku_array[i][j]=returnstr(display_EditText_array[ctr++].getText().toString());

       return;
    }
    public void check1(int ids1)
    {   int i2=0;
        int xstart=0,ystart=0;
       int cellcode=ids1-cell_id;
       int xx1=(int)(cellcode/9);
       int xx2=cellcode-(9*xx1);
       for(int i1=0;i1<10;i1++)
             array_button_dialog[i1]=0;

        i2=xx1;
        for(int j=0;j<9;j++)
             array_button_dialog[sudoku_array[i2][j]]=1;

        i2=xx2;
        for(int j=0;j<9;j++)
             array_button_dialog[sudoku_array[j][i2]]=1;

        if(xx1<3) xstart=0;
        if(xx1>2&&xx1<6) xstart=3;
        if(xx1>5) xstart=6;
        if(xx2<3) ystart=0;
        if(xx2>2&&xx2<6) ystart=3;
        if(xx2>5) ystart=6;

        for(int i=xstart;i<xstart+3;i++)
             for(int j=ystart;j<ystart+3;j++)
                 array_button_dialog[sudoku_array[i][j]]=1;

        return;
     }

    public void setup_edittext_listener()
    {  final AbsoluteLayout a1=(AbsoluteLayout) findViewById(R.id.abs2);
        for(int i=0;i<81;i++)
        {   v2 = a1.getChildAt(i);
            if(v2 instanceof EditText)
            {   EditText a2=(EditText) v2;
                a2.setOnClickListener(new View.OnClickListener() {
```

```
                    @Override
                    public void onClick(View view) {
                    int ids=view.getId();
                    check1(ids);
                    numBoard(ids);  }     });   }     }
     return;
  }

public void display()
{  int ctr=0;
    for(int i=0;i<9;i++)
         for(int j=0;j<9;j++)
            display_textview_array[ctr++].setText(String.valueOf(sudoku_array[i][j]));

    return;
   }

public void start()
{  for(int i=0;i<9;i++)
       for(int j=0;j<9;j++)
            sudoku_array[i][j]=0;

    setup_edittext_listener();
   Button button1=(Button)findViewById(R.id.button_solve);
         button1.setOnClickListener(new OnClickListener()
         {     @Override
            public void onClick(View v)
               {
                 start_solve();
         }});

    Button button2=(Button)findViewById(R.id.reset);
         button2.setOnClickListener(new OnClickListener()
         {     @Override
               public void onClick(View v)
               {
               reset(); }});
         Button quit=(Button) findViewById(R.id.bback1);
    quit.setOnClickListener(new OnClickListener()
         {   @Override
               public void onClick(View v) {
               quit_ask();

            }});
    return;
         }
public void quit_ask()
{
   AlertDialog.Builder builder = new AlertDialog.Builder(this);
           builder.setMessage("Back, are you sure?")
         .setCancelable(false)
         .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
             public void onClick(DialogInterface dialog, int id) {
                 finish();
             }
```

```
            })
            .setNegativeButton("No", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                     dialog.cancel();
                }
            });

              AlertDialog alert = builder.create();
              alert.show();
              return;
}

public void dialog_function_display()
{  final AbsoluteLayout a3=(AbsoluteLayout) findViewById(R.id.abs2);
   int diff=id1-cell_id;
   TextView temp=(TextView) a3.getChildAt(diff);
   if(to_return!=0) temp.setText(String.valueOf(to_return));
   if(to_return==0) temp.setText(String.valueOf(""));
   capturescreen();
   return;
}

public void numBoard(int id2)
{        id1=id2;
   final Context mContext = this;
   final Dialog dialog = new Dialog(mContext);
          dialog.setContentView(R.layout.keyboard);
   dialog.setTitle("NumBoard");
   dialog.show();

   numarray_button[0]=(Button) dialog.findViewById(R.id.num0);
   numarray_button[1]=(Button) dialog.findViewById(R.id.num1);
   numarray_button[2]=(Button) dialog.findViewById(R.id.num2);
   numarray_button[3]=(Button) dialog.findViewById(R.id.num3);
   numarray_button[4]=(Button) dialog.findViewById(R.id.num4);
   numarray_button[5]=(Button) dialog.findViewById(R.id.num5);
   numarray_button[6]=(Button) dialog.findViewById(R.id.num6);
   numarray_button[7]=(Button) dialog.findViewById(R.id.num7);
   numarray_button[8]=(Button) dialog.findViewById(R.id.num8);
   numarray_button[9]=(Button) dialog.findViewById(R.id.num9);

   for(int h=1;h<10;h++)
          if(array_button_dialog[h]==1)
                numarray_button[h].setEnabled(false);

   for(int h=1;h<10;h++)
          if(array_button_dialog[h]==0)
                numarray_button[h].setBackgroundResource(R.drawable.b_assist);

   numarray_button[1].setOnClickListener(new View.OnClickListener()
       {  @Override
            public void onClick(View view)
            {   to_return=1;
                dialog_function_display();
                dialog.dismiss();         }        });
```

```
numarray_button[2].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=2;
       dialog_function_display();
       dialog.dismiss();         }      });

numarray_button[3].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=3;
       dialog_function_display();
       dialog.dismiss();      }    });

numarray_button[4].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=4;
       dialog_function_display();
       dialog.dismiss();  }     });

numarray_button[5].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=5;
       dialog_function_display();
       dialog.dismiss(); }       });

numarray_button[6].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=6;
       dialog_function_display();
       dialog.dismiss();   }     });

numarray_button[7].setOnClickListener(new View.OnClickListener()
{  @Override
      public void onClick(View view)
      {       to_return=7;
       dialog_function_display();
       dialog.dismiss();    }    });

numarray_button[8].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=8;
       dialog_function_display();
       dialog.dismiss();           }   });

numarray_button[9].setOnClickListener(new View.OnClickListener()
{   @Override
      public void onClick(View view)
      {       to_return=9;
       dialog_function_display();
```

```
            dialog.dismiss();         }            });

      numarray_button[0].setBackgroundResource(R.drawable.b_eraser);
      numarray_button[0].setOnClickListener(new View.OnClickListener()
      {   @Override
            public void onClick(View view)
            {        to_return=0;
              dialog_function_display();
              dialog.dismiss();         }            });

      Button back1=(Button) dialog.findViewById(R.id.numback1);
      back1.setBackgroundResource(R.drawable.b_x);
      back1.setOnClickListener(new View.OnClickListener()
      {   @Override
            public void onClick(View view)
            {
             dialog.dismiss();    }     });

      Button back=(Button) dialog.findViewById(R.id.numback);
      back.setEnabled(false);

      return ;
       }

    public void reset()
    {        setContentView(R.layout.sudoku_solver);
      setup_display_textview();
      start();
      return;
    }

    public void start_solve()
    {  run_solve();
        final  ProgressDialog dialog = ProgressDialog.show(Sudoku_solver.this, "",
"Solving.", true);
        new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
             dialog.dismiss();
             display();
           }
    }, 2000);

            return;
    }

    /*SUDOKU SOLVER*/

    public void run_solve()
    {  try
       {   start_finding( 0, 0 ) ;     }
       catch( Exception e )   {   }
    }

    public  void next_cell_find( int r, int c ) throws Exception
```

```
    {  if( c < 8 )  start_finding( r, c + 1 ) ;
       else         start_finding( r + 1, 0 ) ;
    }

    public void start_finding( int r, int c ) throws Exception
    {   if( r > 8 ){
             throw new Exception( "Solution found" ) ;     }
         if(sudoku_array[r][c] != 0 )
          next_cell_find( r, c ) ;
       else
       {  for( int num = 1; num < 10; num++ )
          {  if( checkRow(r,num) && checkCol(c,num) && checkBox(r,c,num) )
             { sudoku_array[r][c] = num ;
               next_cell_find( r, c ) ;      }    }
        sudoku_array[r][c] = 0 ;    }
    }

    protected boolean checkBox( int row, int col, int num )
    {  row = (row / 3) * 3 ;
       col = (col / 3) * 3 ;
      for( int r = 0; r < 3; r++ )
          for( int c = 0; c < 3; c++ )
          if(sudoku_array[row+r][col+c] == num )
             return false ;

       return true ;
    }

    protected boolean checkRow( int row, int num )
    {  for( int col = 0; col < 9; col++ )
          if( sudoku_array[row][col] == num )
             return false ;

       return true ;
    }

    protected boolean checkCol( int col, int num )
    {  for( int row = 0; row < 9; row++ )
          if( sudoku_array[row][col] == num )
             return false;

       return true ;
    }

    public int returnstr(String input1)
    {  if(input1.length()==0) return 0;
       else return Integer.parseInt(input1);
    }
}
```

7.Credit

```
package com.sarath.sv17;

import android.app.Activity;
import android.app.Dialog;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.RelativeLayout;

public class Credit extends Activity{

        @Override
    public void onCreate(Bundle savedInstanceState)
        {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.credit);
        Button sv1=(Button)findViewById(R.id.sv1);
        sv1.setOnClickListener(new View.OnClickListener()
           {
               public void onClick(View view)
               {
                     display(1);
                }
             });

        Button sv2=(Button)findViewById(R.id.sv2);
        sv2.setOnClickListener(new View.OnClickListener()
           {
               public void onClick(View view)
               {
                     display(2);
                }
             });
        Button vs1=(Button)findViewById(R.id.vs1);
       vs1.setOnClickListener(new View.OnClickListener()
           {
               public void onClick(View view)
               {
                     display(3);
                }
             });

        Button back=(Button)findViewById(R.id.b_back);
        back.setOnClickListener(new View.OnClickListener()
           {
               public void onClick(View view)
               {
                           finish();
                }
             });
       }
```

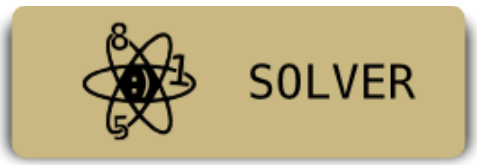
```
    public void display(int a)
    {
            final Context mContext = this;
        final Dialog dialog = new Dialog(mContext);
              dialog.setContentView(R.layout.sv1);
              RelativeLayout rl1=(RelativeLayout) dialog.findViewById(R.id.rlsv);
        if(a==1) rl1.setBackgroundResource(R.drawable.bg_sarath);
        if(a==2) rl1.setBackgroundResource(R.drawable.bg_sujith);
        if(a==3) rl1.setBackgroundResource(R.drawable.bg_vig);

        rl1.setOnClickListener(new View.OnClickListener()
            {
                public void onClick(View view)
                {
                            dialog.dismiss();
                 }
               });
        dialog.show();
        return;
    }


}
```

Graphics

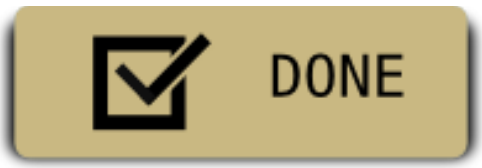

NEW
GAME


SOLVER

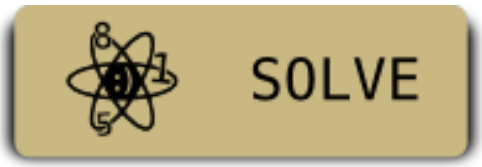

CREDITS


EXIT


ACTION


DONE


PAUSE


SOLVE


RESET

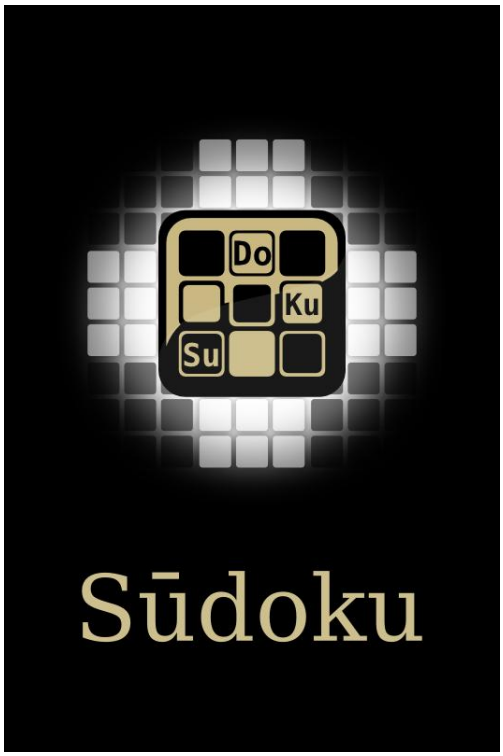

Sarath
Vadakkepat


Sujith
Vadakkepat

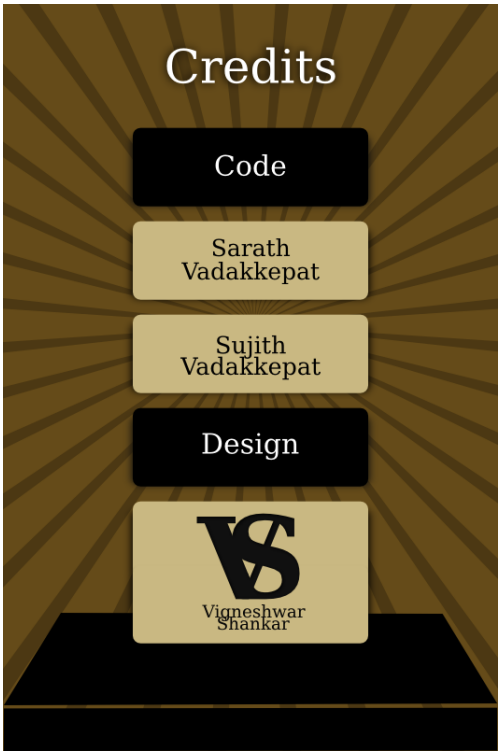

Vigneshwar
Shankar

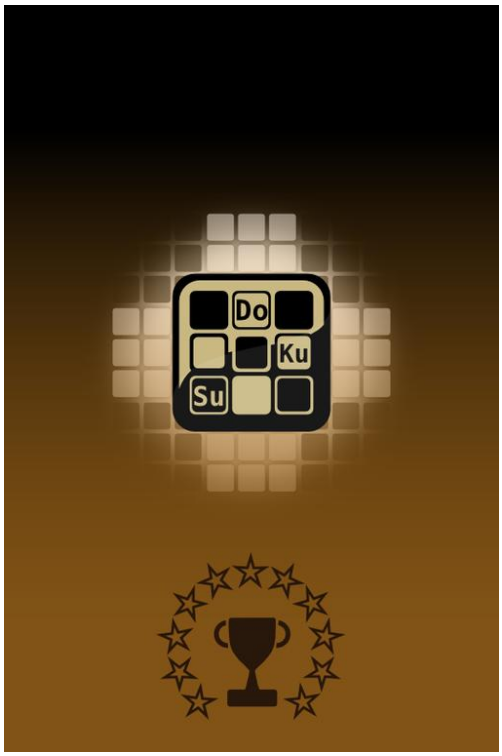

Code


Design


Do
Ku
Su
Sūdoku


NEW GAME
SOLVER
CREDITS
EXIT

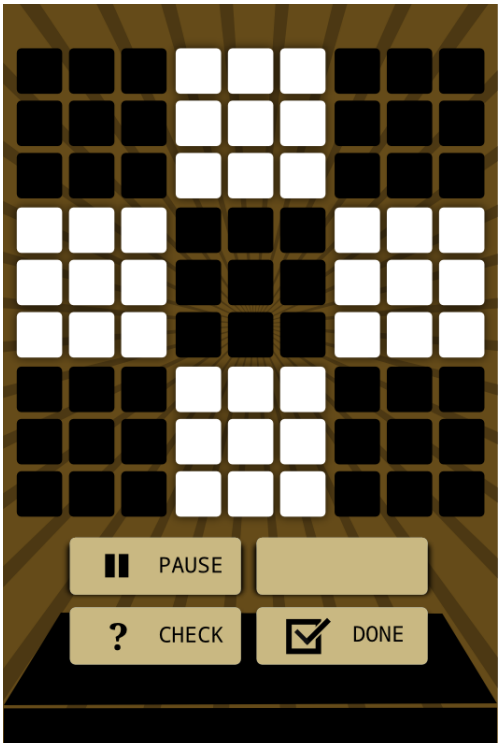

Credits
Code
Sarath Vadakkepat
Sujith Vadakkepat
Design
Vigneshwar Shankar

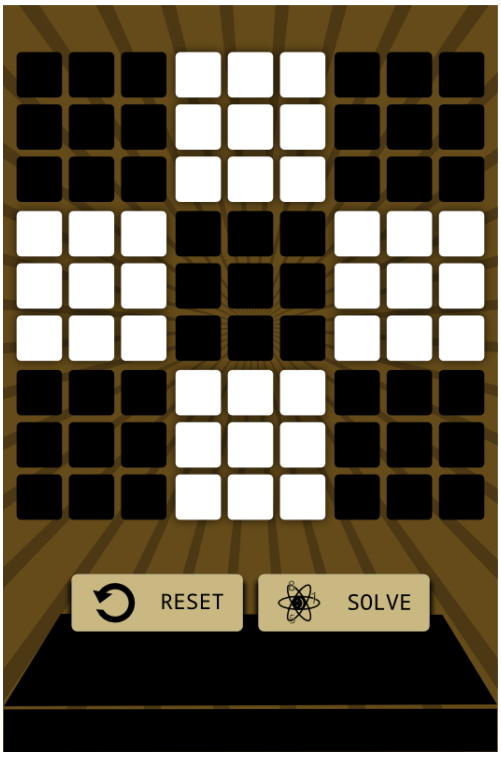

Do
Ku
Su

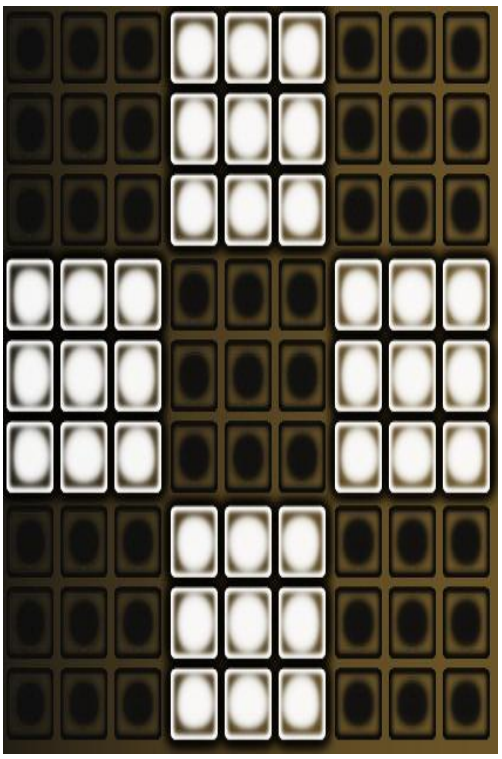

Sarath
Vadakkepat
@srthvadakkepat
about.me/sarathvadakkepat

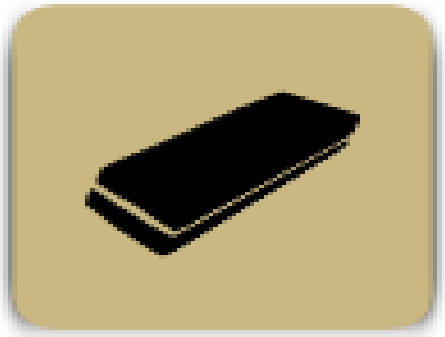

Sujith
Vadakkepat
fb.me/sujith.vadakkepat

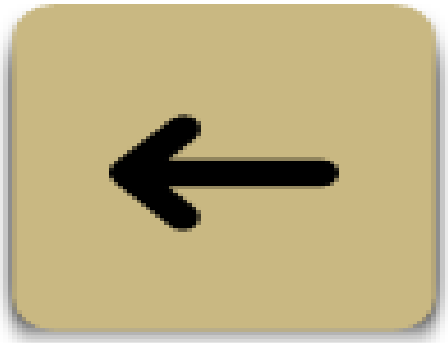

Vigneshwar
Shankar
@vgnshwr
about.me/vigneshwar


PAUSE
CHECK
DONE


RESET
SOLVE







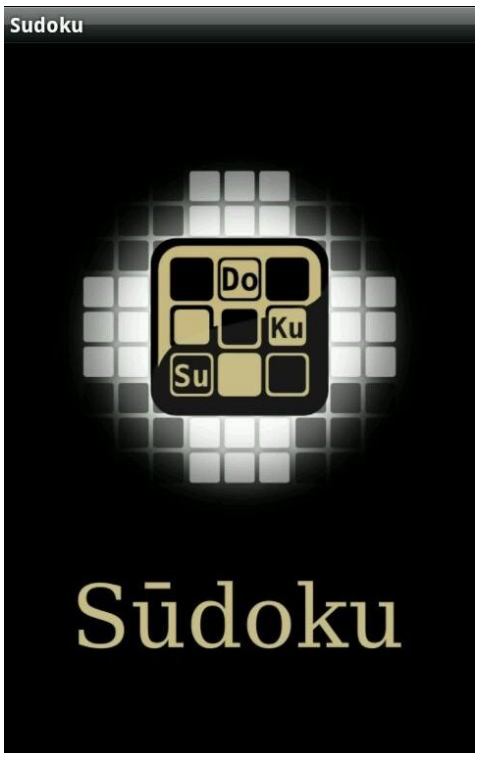

Do
Ku
Su

Screen Shots

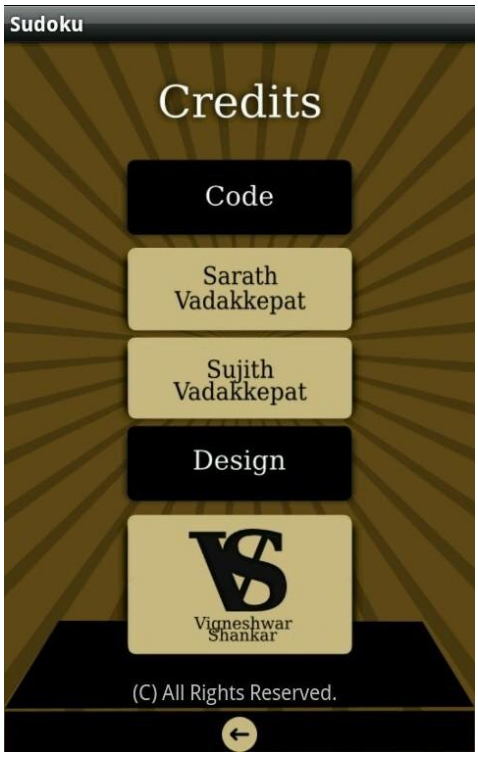
Taken From an actual Device

Device Name : HTC Desire S
Manufacturer : HTC
Screen Size : 3.7”
OS Version : v2.3 Gingerbread
Date : 19th July, 2012

Courtesy : Prahlad Suresh


Sudoku
Do
Ku
Su
Sūdoku


Sudoku
NEW GAME
SOLVER
CREDITS
EXIT


Sudoku
Credits
Code
Sarath Vadakkepat
Sujith Vadakkepat
Design
Vigneshwar Shankar
(C) All Rights Reserved.

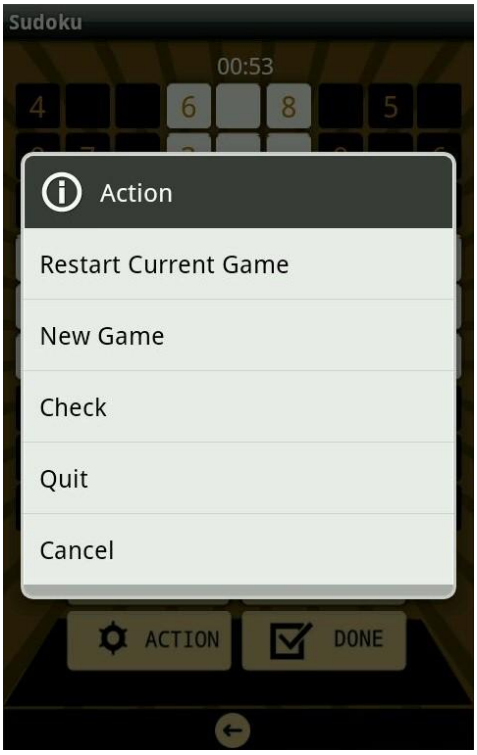

Sarath Vadakkepat
@srthvadakkepat
about.me/sarathvadakkepat


Sujith
Vadakkepat
fb.me/sujith.vadakkepat

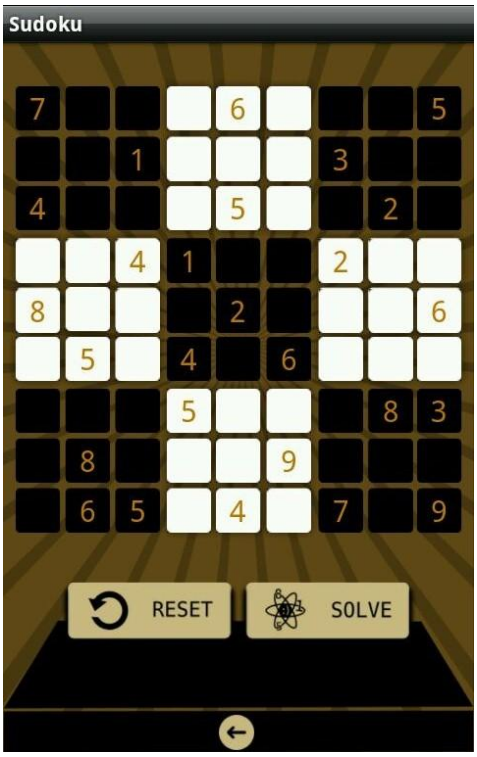

Vigneshwar
Shankar
@vgnshwr
about.me/vigneshwar


Sudoku
Back, are you sure?
Yes
No
RESET
SOLVE

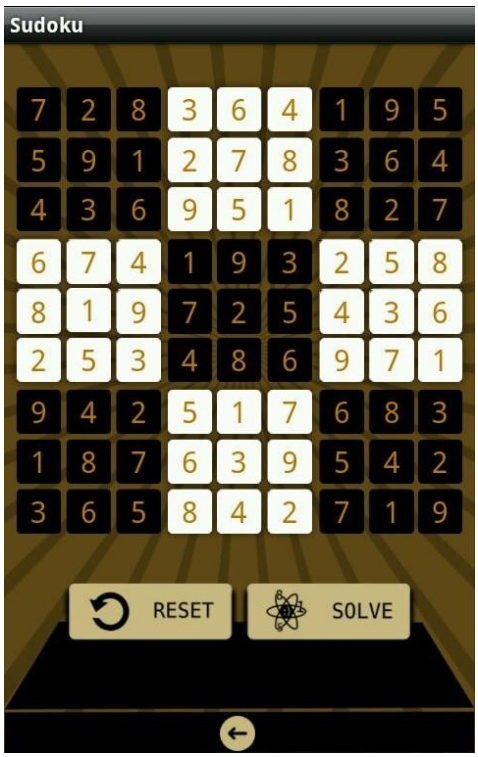

Sudoku
Level
Easy
Medium
Hard

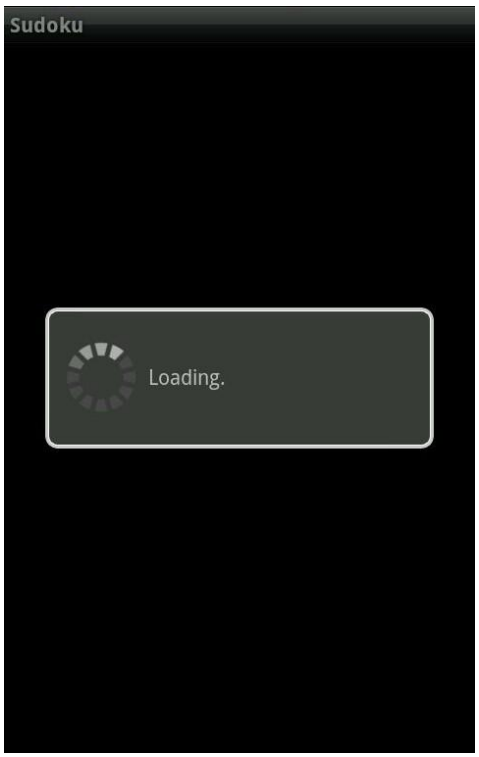

Sudoku
00:53
Action
Restart Current Game
New Game
Check
Quit
Cancel
ACTION
DONE


Sudoku
01:00
Restart, are you sure?
Yes
No
PAUSE
ASSIST
ACTION
DONE

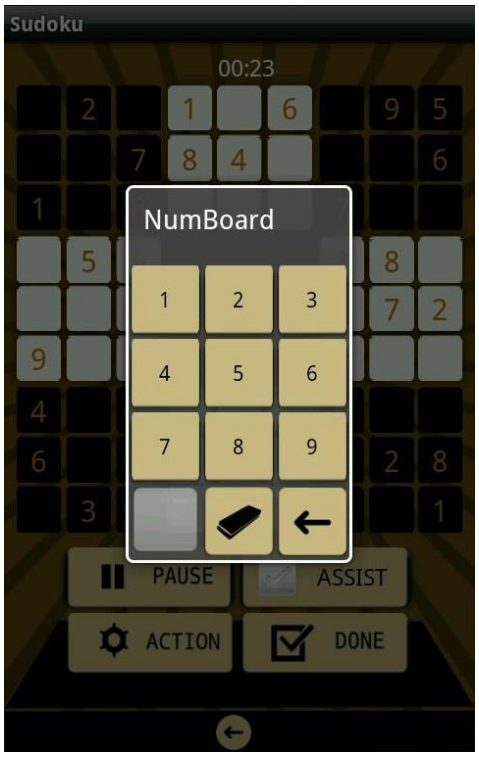

Sudoku
RESET
SOLVE

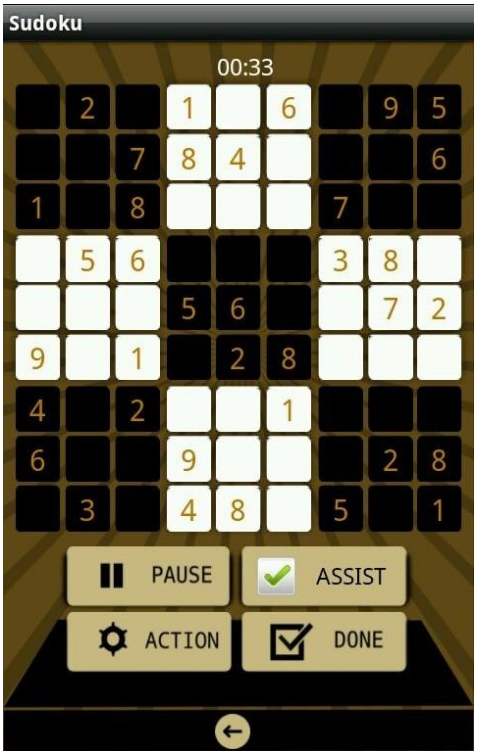

Sudoku
Solving.
RESET
SOLVE

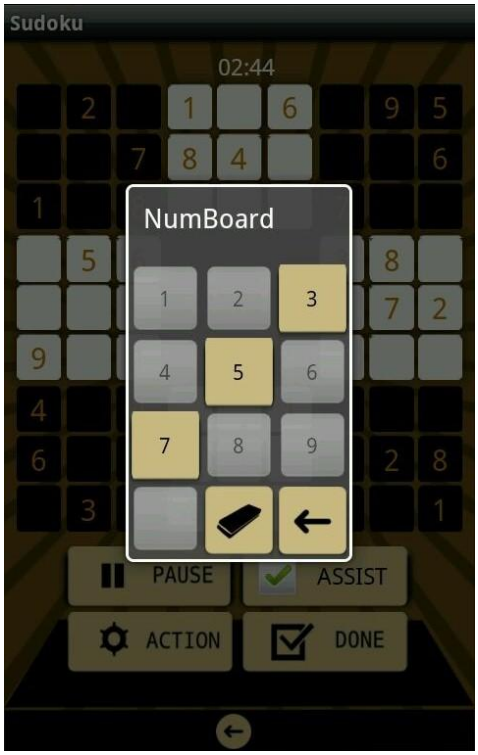

Sudoku
7 2 8 3 6 4 1 9 5
5 9 1 2 7 8 3 6 4
4 3 6 9 5 1 8 2 7
6 7 4 1 9 3 2 5 8
8 1 9 7 2 5 4 3 6
2 5 3 4 8 6 9 7 1
9 4 2 5 1 7 6 8 3
1 8 7 6 3 9 5 4 2
3 6 5 8 4 2 7 1 9
RESET
SOLVE

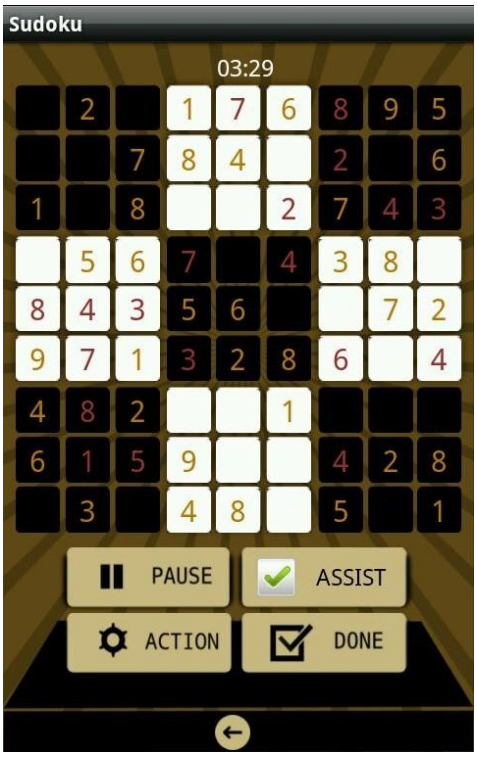

Sudoku
Loading.


Sudoku
00:12
PAUSE
ASSIST
ACTION
DONE


Sudoku
00:23
NumBoard
1 2 3
4 5 6
7 8 9
PAUSE
ASSIST
ACTION
DONE


Sudoku
00:33
PAUSE
ASSIST
ACTION
DONE

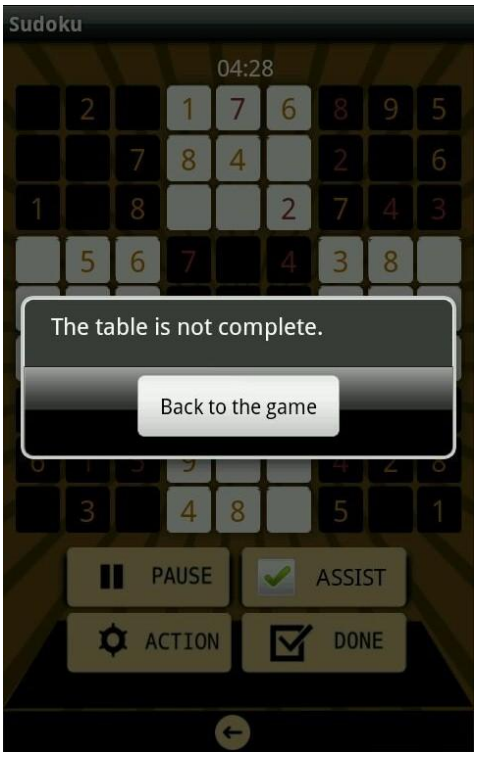

Sudoku
02:44
NumBoard
1
2
3
4
5
6
7
8
9
PAUSE
ASSIST
ACTION
DONE

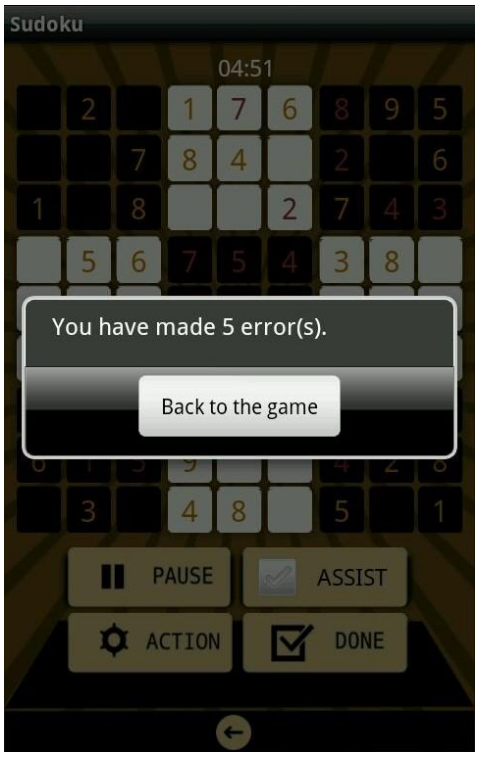

Sudoku
03:29
PAUSE
ASSIST
ACTION
DONE

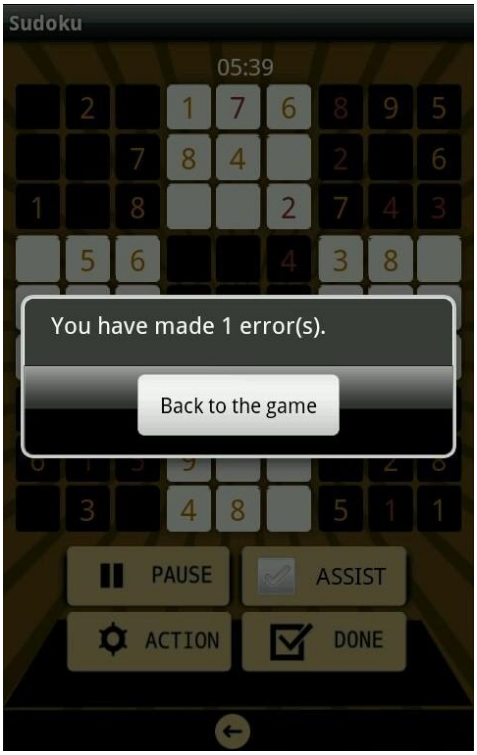

Sudoku
03:46
New Game, are you sure?
Yes
No
PAUSE
ASSIST
ACTION
DONE


Sudoku
04:12
Quit, are you sure?
Yes
No
PAUSE
ASSIST
ACTION
DONE


Sudoku
04:21
So far so good.
Back to the game
PAUSE
ASSIST
ACTION
DONE

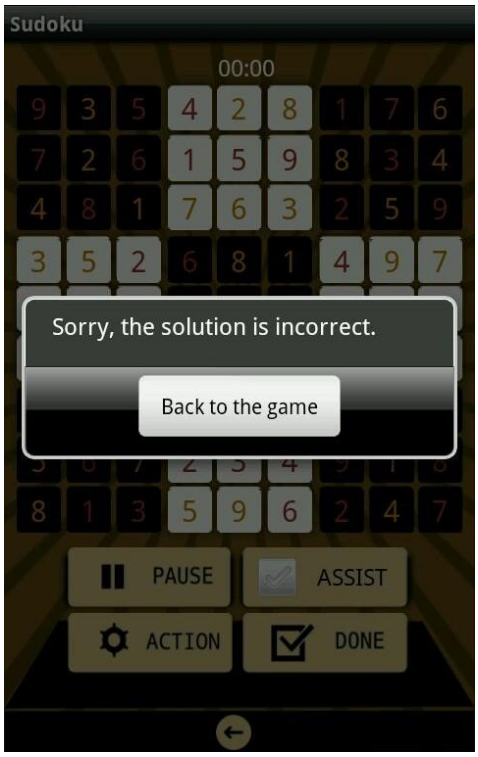

Sudoku
04:28
The table is not complete.
Back to the game
PAUSE
ASSIST
ACTION
DONE


Sudoku
04:51
You have made 5 error(s).
Back to the game
PAUSE
ASSIST
ACTION
DONE

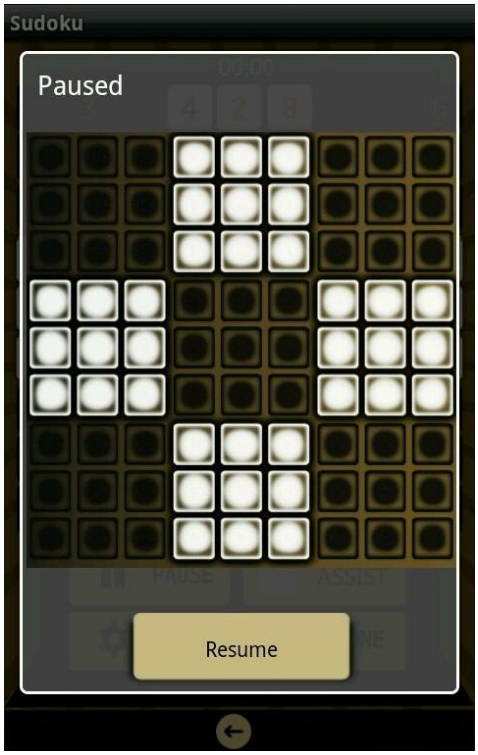

Sudoku
05:39
You have made 1 error(s).
Back to the game
PAUSE
ASSIST
ACTION
DONE


Congratulations
You have solved the puzzle in
2 Minute(s) 14 Second(s).
Do
Ku
Su


Sudoku
00:00
Next is what?
Main menu
New game
Cancel
PAUSE
ASSIST
ACTION
DONE


Sudoku
00:00
Sorry, the solution is incorrect.
Back to the game
PAUSE
ASSIST
ACTION
DONE


Sudoku
00:03
Back, are you sure?
Yes
No
PAUSE
ASSIST
ACTION
DONE


Sudoku
Paused
Resume

References

The project was developed mainly using the official android documentation

http://developer.android.com/guide/components/index.html

Various tech blogs depicting the use of codes in sample programs were referred for guidance.

Developer Page

Code:

Sarath Vadakkepat
@srthvadakkepat
sarath.vadakkepat@gmail.com

Graphics:

Vigneshwar Shankar
@vgnshwar
vigneshwar.s@gmail.com


Do
Ku
Su