CHECKERS GAME TOURNAMENT

OPERATING SYSTEM PROJECT REPORT

MARCH 2, 2022

ABDUL BASIT
MARYAM AHMED

ABOUT THE PROJECT:

Checkers game tournament with multiple games running at the same time. A record of players is maintained on a MYSQL server.

FEATURES:

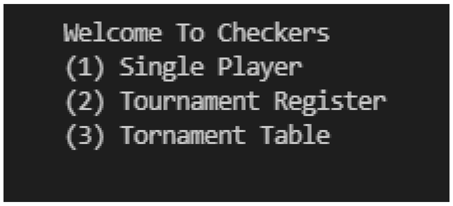
- Multithreading
- Tournament
- Database connection
- Java Code
- Play against computer
- Socket Programming

TOOLS AND TECHNOLOGIES:

- Using JAVA application to create checkers game
- Implementing Socket server concepts to establish a connection between two instances of the game to allow two players to compete with each other
- Implementing MYSQL server to create and maintain a database of all players and games which links to the JAVA application of the checkers game
- Implementing multithreading concepts to run multiple games at the same time

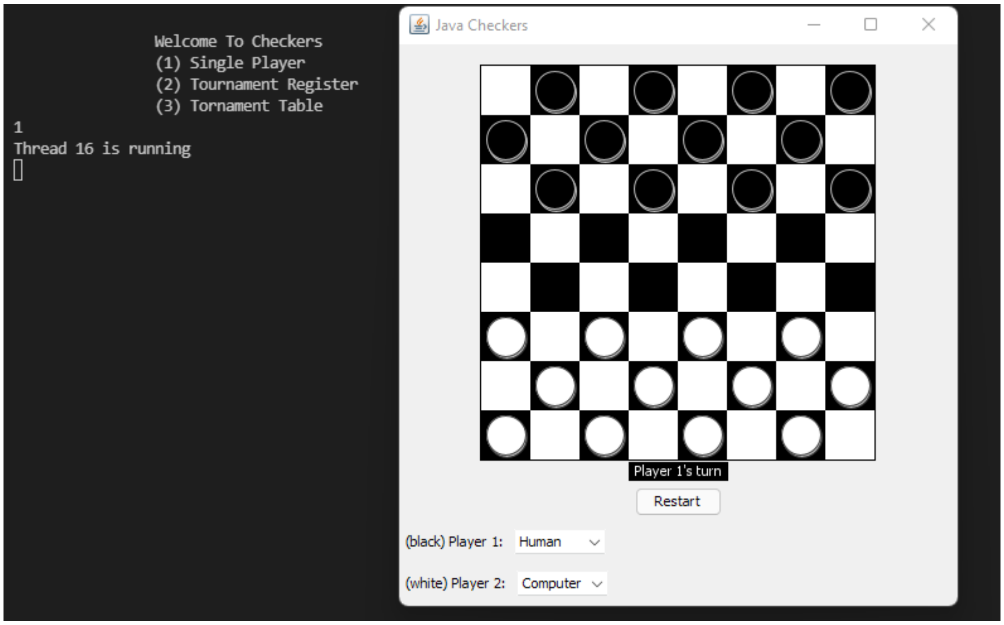
PROJECT FLOW:

- Games starts and gives choice for single player, Tournament registration or tournament table.



```
  Welcome To Checkers
  (1) Single Player
  (2) Tournament Register
  (3) Tornament Table
```

- <u>Choosing 1</u> will generate a game running on a new thread for single player
  - User can play against computer
  - User can play against another human
  - User can play against any other player using IP

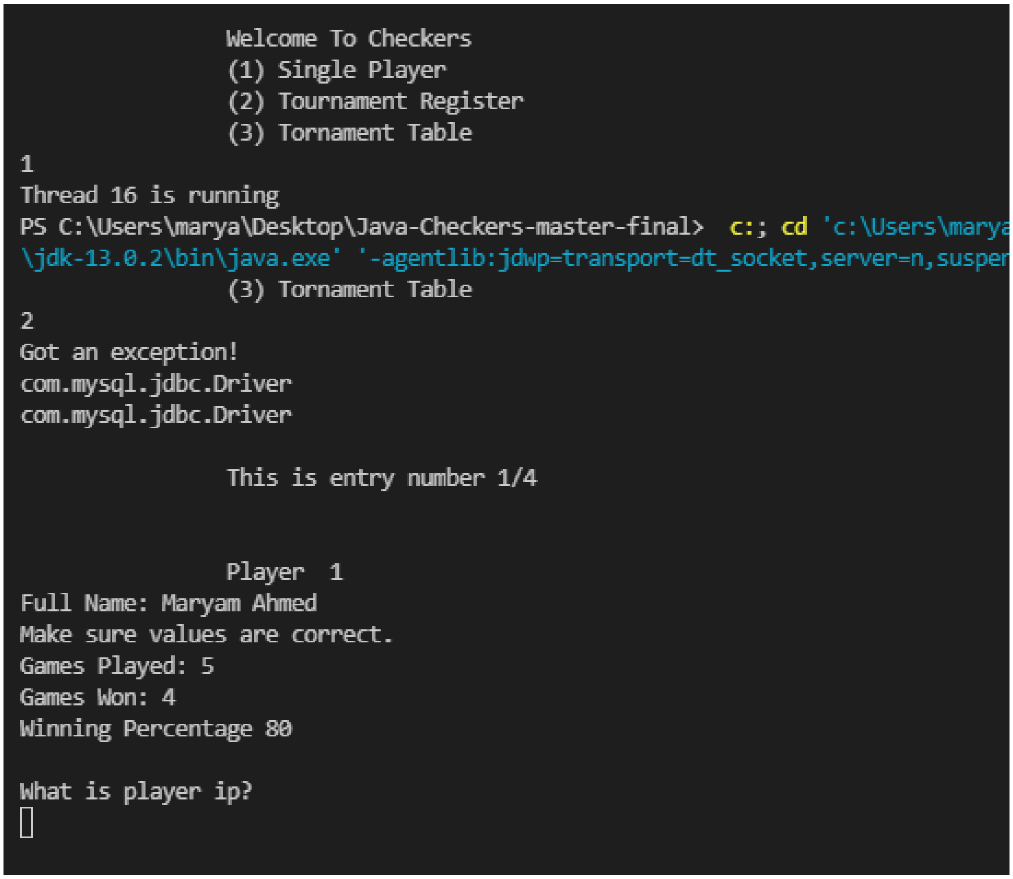


```
              Welcome To Checkers
              (1) Single Player
              (2) Tournament Register
              (3) Tornament Table
1
Thread 16 is running
▯
```

(black) Player 1: Human

(white) Player 2: Computer

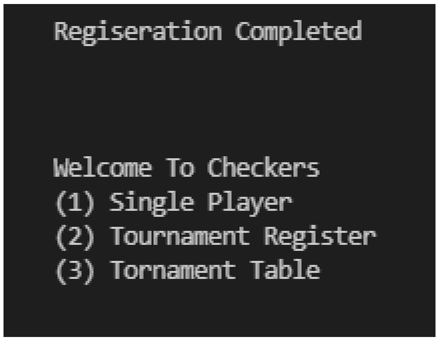
- <u>Choosing 2</u> will open registration screen which will take input of:
  - **Full name**: We will use the name to check whether the user is registered for the tournament
  - **Number of games played/ games won**: This will be used to generate win percentage of the player **(Program will verify values and generate winning percentage itself)**
  - **Players IP:** This will be used to connect two users



```
                Welcome To Checkers
                (1) Single Player
                (2) Tournament Register
                (3) Tornament Table
1
Thread 16 is running
PS C:\Users\marya\Desktop\Java-Checkers-master-final>  c:; cd 'c:\Users\marya
\jdk-13.0.2\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,susper
                (3) Tornament Table
2
Got an exception!
com.mysql.jdbc.Driver
com.mysql.jdbc.Driver

                This is entry number 1/4


                Player  1
Full Name: Maryam Ahmed
Make sure values are correct.
Games Played: 5
Games Won: 4
Winning Percentage 80

What is player ip?
```

- **All this data will be stored in remotely connected MYSQL Server so that anyone with the application can access it.**
- **Before registering data our program will also check from the MYSQL whether or not the registration limit is exceeded.**

After entering IP, registration completed and back to main menu

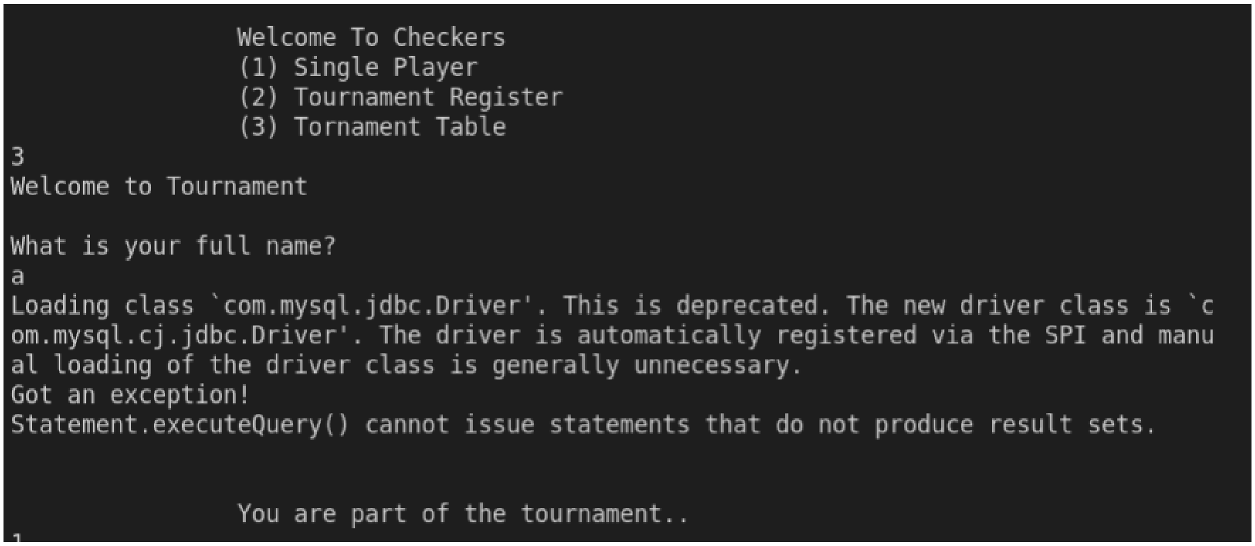


```
Regiseration Completed



Welcome To Checkers
(1) Single Player
(2) Tournament Register
(3) Tornament Table
```

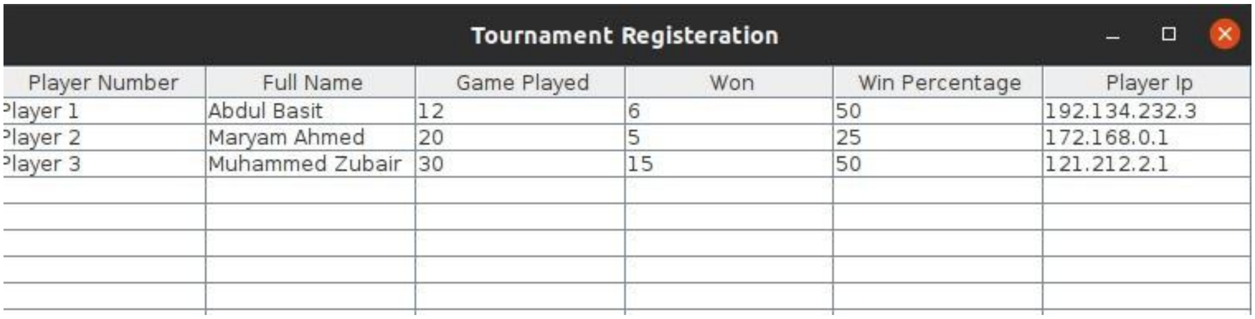
- Choosing 3:

  Will ask for the users full name **It will verify** from the MYSQL database whether or not the user exists in the registration.

  If it does, tournament details such as opponents IP and name will be displayed on a J-table



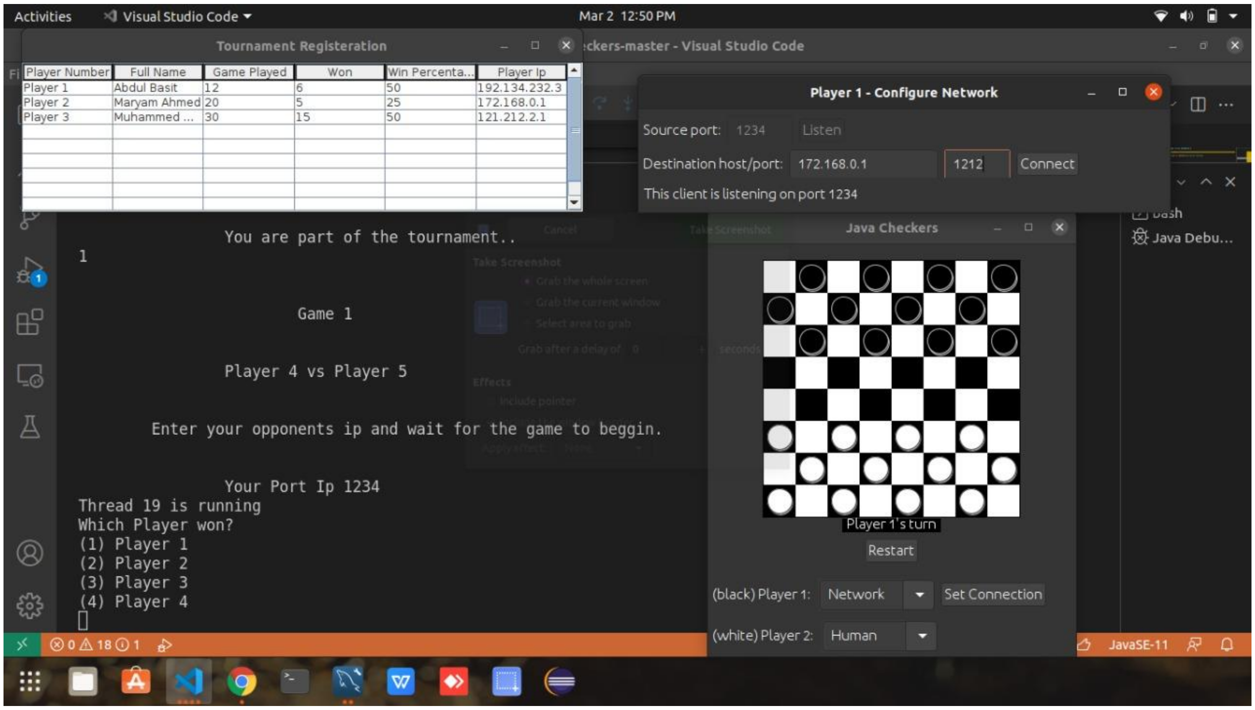
```
                Welcome To Checkers
                (1) Single Player
                (2) Tournament Register
                (3) Tornament Table
3
Welcome to Tournament

What is your full name?
a
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `c
om.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manu
al loading of the driver class is generally unnecessary.
Got an exception!
Statement.executeQuery() cannot issue statements that do not produce result sets.


                You are part of the tournament..
```



Tournament Registeration

Player Number	Full Name	Game Played	Won	Win Percentage	Player Ip
Player 1	Abdul Basit	12	6	50	192.134.232.3
Player 2	Maryam Ahmed	20	5	25	172.168.0.1
Player 3	Muhammed Zubair	30	15	50	121.212.2.1

- Now player can connect to the opponent using the IP displayed on the J-table.



Tournament Registeration

Player Number	Full Name	Game Played	Won	Win Percenta...	Player Ip
Player 1	Abdul Basit	12	6	50	192.134.232.3
Player 2	Maryam Ahmed	20	5	25	172.168.0.1
Player 3	Muhammed ...	30	15	50	121.212.2.1

```
                    You are part of the tournament..
 1


                              Game 1


                    Player 4 vs Player 5


          Enter your opponents ip and wait for the game to beggin.


                    Your Port Ip 1234
Thread 19 is running
Which Player won?
(1) Player 1
(2) Player 2
(3) Player 3
(4) Player 4
```

Player 1 - Configure Network

Source port: 1234 Listen

Destination host/port: 172.168.0.1 1212 Connect

This client is listening on port 1234

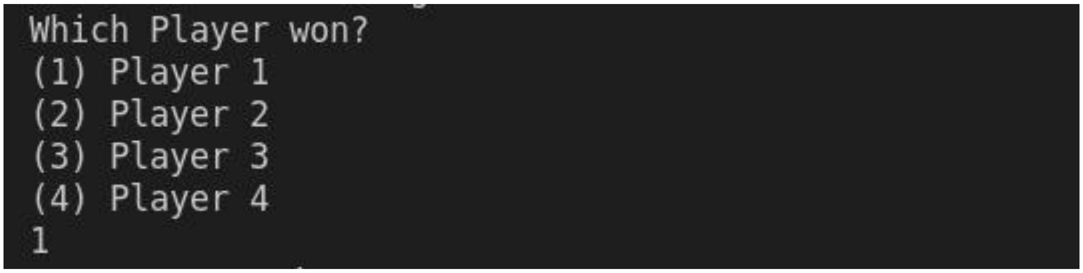
Java Checkers

Player 1's turn

(black) Player 1: Network

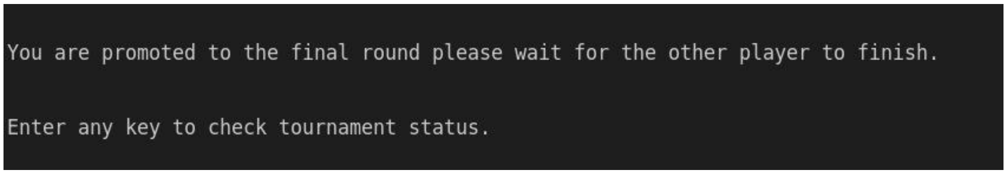
(white) Player 2: Human

- After the game ends, the table will be updated and the looser will be eliminated from the tournament table.



```
Which Player won?
(1) Player 1
(2) Player 2
(3) Player 3
(4) Player 4
1
```

- The player won will be promoted and wait for the other player to finish the match
- J-table will display the status of the tournament and after the completion of both the players game a new instance will be created displaying players info and tournament details.



```
You are promoted to the final round please wait for the other player to finish.


Enter any key to check tournament status.
```

<u>CODE</u>

MENU

```
package ui;

import java.io.IOException; //throw a failure in Input & Output
import java.util.Scanner;
public class Menu {
   public static void main(String[] args) throws Exception {
      Scanner scan= new Scanner(System.in);
      System.out.println("\n\n\t\tWelcome To Checkers");//print function
      System.out.println("\t\t(1) Single Player\n\t\t(2) Tournament Register\n\t\t(3) Tornament Table");
      int num = scan.nextInt();
      if (num  == 1){
         Start st = new Start();
        Start.main(args);
      }
      if (num  == 2){
         Registeration reg = new Registeration();
         reg.main(args);
      }
      if (num  == 3){
         Tournament Tour = new Tournament();
         Tour.main(args);
      }
   }
}
```

REGISTRATION

```
package ui;
import java.io.*;

public class Registeration {
public static <Mysql> void main(String[] args) throws Exception
 {
      //create array
      String[][] Data = new String[10][10]; // create 5 columns, 5 rows
      BufferedReader bfn= new BufferedReader(new InputStreamReader(System.in));
      for (int row = 0; row < Data.length; row++)
   {
      Sql_Extract check = new Sql_Extract(0,0);
      int limit=check.Count_Rows();
      if (limit>=4){
         System.out.println("\n\n\n\t\tSorry registerations are closed. \n\t\t    Wait for next
tournament.\n\nPress any key to continues");
         System.in.read();
         Menu st = Menu();
         st.main(args);
        }
      System.out.println("\n\t\tThis is entry number "+(limit+1)+"/"+4);
      System.out.println("\n\n\t\tPlayer  "+(limit+1));
      Data[row][0] = "Player "+(limit+1);
      System.out.print("Full Name: ");
      Data[row][1] = bfn.readLine();
      int won;
      int played;
      do{
         System.out.print("Make sure values are correct.\n");
         System.out.print("Games Played: ");
         Data[row][2] = bfn.readLine();

         System.out.print("Games Won: ");
         Data[row][3] = bfn.readLine();
         won = Integer.parseInt(Data[row][3]);
         played = Integer.parseInt(Data[row][2]);
      }while(won >played);///check
      int win_per = (won*100/played);
      System.out.print("Winning Percentage "+win_per);
      Data[row][4] = String.valueOf(win_per);
      System.out.print("\n\nWhat is player ip?\n ");
      Data[row][5] = bfn.readLine();
      /*
      JTableExamples jt = new JTableExamples(Data);
      jt.main(Data);
      */
      Sql_Insert sql = new Sql_Insert(Data) ;
      sql.main(Data);
      Sql_Extract sq= new Sql_Extract(0,1);
      sq.main(0,1);
      System.out.print("\n\t\tRegiseration Completed\n\n");
      Menu mai = new Menu();
      mai.main(args);
   }
 }
private static Menu Menu() {
   return null;
}
}
```

SQL EXTRACT

```
package ui;
import java.sql.*;
import com.mysql.cj.Query;
import com.mysql.cj.protocol.Resultset;
public class Sql_Extract
{
 void Delete_Player(int player_number)
 {
        String query=" ";
        try{
             Class.forName("com.mysql.jdbc.Driver");
             Connection con=DriverManager.getConnection(
             "jdbc:mysql://localhost:3306/Mysql","admin","abc123");
             //here sonoo is database name, root is username and password
              // create the java statement
              Statement st = con.createStatement();
              // our SQL SELECT query.
             // if you only need a few columns, specify them by name instead of using "*"
               query = "Delete FROM Data where s_no = "+player_number+";";
               System.out.println(query);
               st.executeQuery(query);
              JTableExamples Table = new JTableExamples(Data);
              Table.main(Data);
            }

          catch (Exception e)
            {
             System.err.println("Got an exception!");
             System.err.println(e.getMessage());
            }
 }

 public int Count_Rows()
 {
  int numberRow = 0;
  try{
     Class.forName("com.mysql.jdbc.Driver");
     Connection con=DriverManager.getConnection(
     "jdbc:mysql://localhost:3306/Mysql","admin","abc123");
     //here sonoo is database name, root is username and password

     // create the java statement
     Statement st = con.createStatement();
     String query=" ";
    query = "select count(*) from Data";
    PreparedStatement stt = con.prepareStatement(query);
    ResultSet rs = stt.executeQuery();
    while(rs.next())
    {
       numberRow = rs.getInt("count(*)");
     }

     }catch (Exception ex)
     {
        System.out.println(ex.getMessage());
     }

 return numberRow
};

public int Player_Verify(String name)
{
 int verify = 0;
 try{
    Class.forName("com.mysql.jdbc.Driver");
    Connection con=DriverManager.getConnection(
    "jdbc:mysql://localhost:3306/Mysql","admin","abc123");
    //here sonoo is database name, root is username and password
    // create the java statement
    Statement st = con.createStatement();
    String query=" ";

   query = "select Full_Name from Data where Full_Name like '%"+name+"%'";
   PreparedStatement stt = con.prepareStatement(query);
   ResultSet rs = stt.executeQuery();
   while(rs.next())
   {
     System.out.println("\n\n\t\tYou are part of the tournament..");
      verify =1;

    }catch (Exception ex)
    {
       System.out.println(ex.getMessage());
    }


return verify;
```

```
};
 public String[][] Data = new String[10][10]; // create 3 columns, 10 rows
    Sql_Extract(int player_number,int query_no)
 {
         try{
               Class.forName("com.mysql.jdbc.Driver");
               Connection con=DriverManager.getConnection(
               "jdbc:mysql://localhost:3306/Mysql","admin","abc123");
               //here sonoo is database name, root is username and password
                // create the java statement
                Statement st = con.createStatement();
                // our SQL SELECT query.
               // if you only need a few columns, specify them by name instead of using "*"
               String query=" ";
               if(query_no == 1){
                 query = "SELECT * FROM Data";
               }
               if(query_no==2){
                 query = "SELECT * FROM Data where Player_Number like '%Player "+player_number+"%'";
               }
               // execute the query, and get a java resultset
                ResultSet rs = st.executeQuery(query);
                int row =0;
                while (rs.next())
                {
                     Data[row][0] = rs.getString("Player_Number");
                     Data[row][1] = rs.getString("Full_name");
                     Data[row][2] = rs.getString("Game_played");
                     Data[row][3] = rs.getString("Game_won");
                     Data[row][4] = rs.getString("Win_percentage");
                     Data[row][5] = rs.getString("Player_Ip");
                     row++;
                 }
                JTableExamples Table = new JTableExamples(Data);
                Table.main(Data);
             }
           catch (Exception e)
             {
               System.err.println("Got an exception!");
               System.err.println(e.getMessage());
             }
 }
                   public static void main(String[] args) {
```

```
                 new Sql_Extract(1,2);
                }
                 public void main(int i, int j) {
                 }
}
```

J TABLE EXAMPLE

```
package ui;
// Packages to import
import javax.swing.JFrame; //swing ui libirary
import javax.swing.JScrollPane;
import javax.swing.JTable;
public class JTableExamples {
   // frame
   JFrame f;
   // Table
   JTable j;
   // Constructor
   JTableExamples(String[][] data)
   {
      // Frame initialization
      f = new JFrame();
      // Frame Title
      f.setTitle("Tournament Registeration");
      // Column Names
      String[] columnNames = { "Player Number", "Full Name", "Game Played","Won","Win 
Percentage","Player Ip" };
      // Initializing the JTable
      j = new JTable(data, columnNames);
      j.setBounds(30, 40, 200, 300);

      // adding it to JScrollPane
      JScrollPane sp = new JScrollPane(j);
      f.add(sp);
      // Frame Size
      f.setSize(500, 200);
      // Frame Visible = true
      f.setVisible(true);
   }
   // Driver  method
   public static void main(String[][] data)
   {

   }
```