Introduction

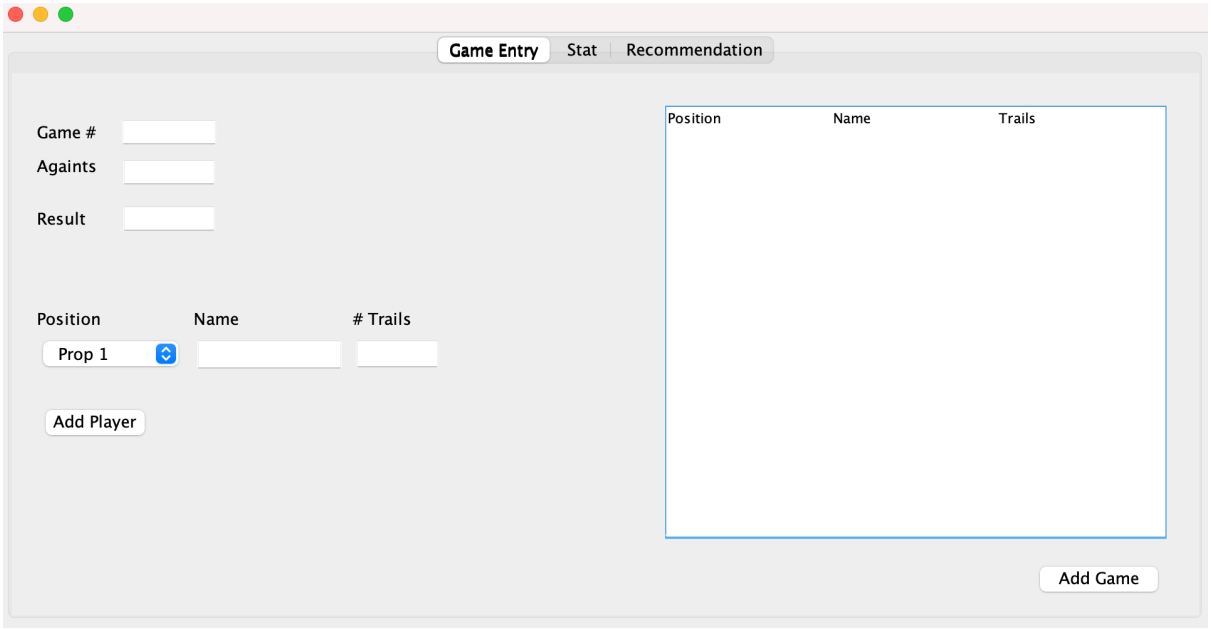
The program, Rugby players predictor, is a database that keeps track of all the important information about the player in the rugby team. The program will run through the information in the database which will then calculate and project the best players for the coach. The process of coding and designing were done through NetBeans.

List of Programming Techniques

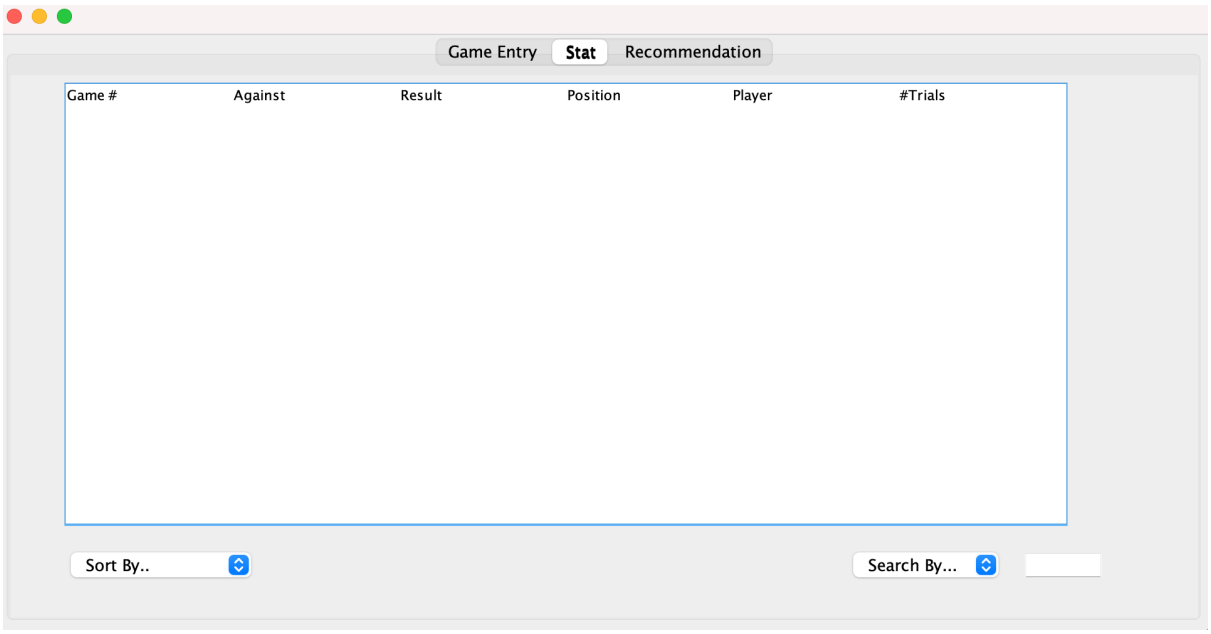
- Graphical user interface (GUI)
- Methods
- Algorithm thinking
- Global and local variables
- for loop
- arrays, 2d arrays
- making an array of objects
- sorting (bubble sort etc.), and in particular sorting an array of objects based on one key attribute
- searching (linear search, binary search…)
- error handling (for example catching a divide by 0 error, or a null pointer while using an array of objects…)
- GUI tabs
- Use of a flag value (such as -999, or "not set yet")

User Interface / GUI work

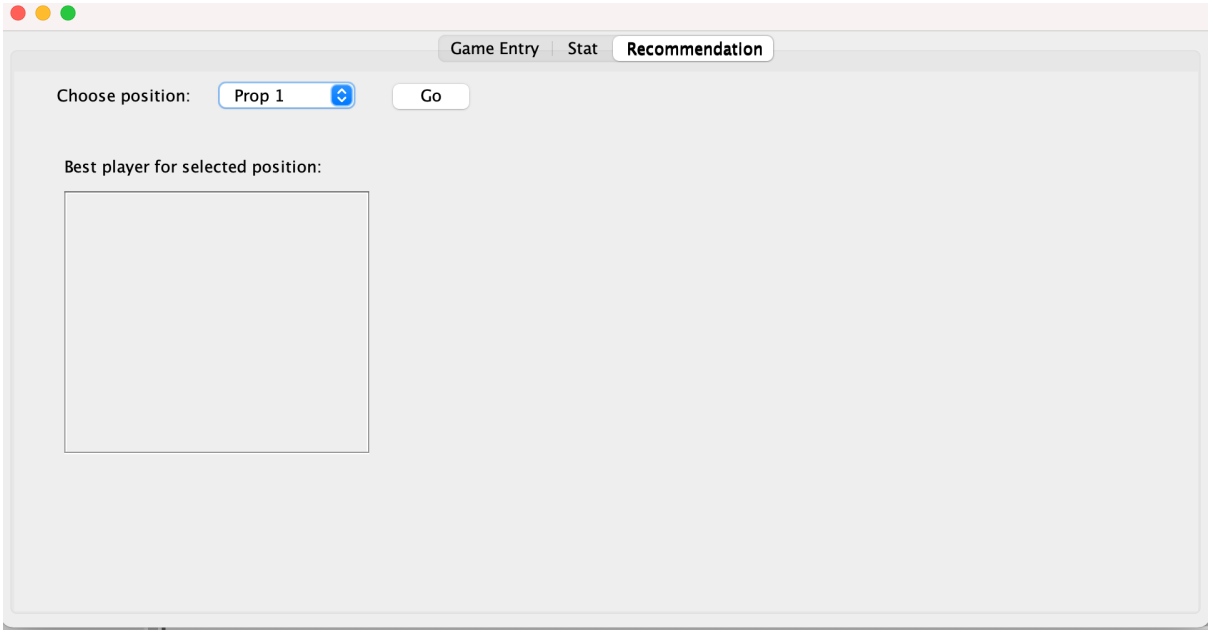
One of the key interface capabilities of Netbeans is the ability to drag and drop Java swing objects into a file, without having to hardcode them. Netbeans allowed all the common features such as, buttons, text pane, dropdown menu, etc to be used easily without having to individually code them.



Common features on the “Game Entry” page are labels, buttons, dropdown menu, table, and textfield. Textfield is used to insert the necessary information for this program to work. The dropdown menu allows users to choose different positions in the rugby line-up. The “add player” button allows users to input the information into the table. Lastly, the “add game” button allows users to input both players’ information and the game information into the table that is located in the “Stat” page.

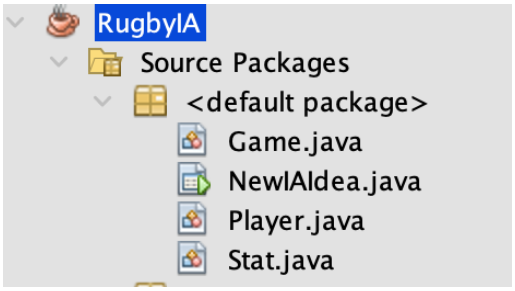


In this “Stat” page, the table presents all the information that is needed by the client. There are two dropdown menus. One is for sorting and the other is for searching. This drop down menu will give the client the freedom to sort and search the table by date, A-Z, positio, etc.



The dropdown menu in the “Recommendations” page let user choose the position for the best overall recommendation. Once the user selects the position in the dropdown menu, the user will have to click the “Go” button to be able see the recommendation. The name of the player will be projected in the text pane.

Structure of the Program



In this program, there are a total of 4 classes. The main class is called “NewIAIdea”. The other classes are “Player”, “Game”, and “Stat”.

/Users/e2120564/NetBeansProjects/RugbyIA/src/Player.java

```
11 public class Player {
12     private String playerPosition = "not yet set";
13     private String playerName = "not yet set";
14     private String playerScore = "not yet set";
15 
16 
17     public Player(){
18 
19     }
20 
21     public Player(String playerPosition, String playerName, String playerScore){
22         this.playerPosition = playerPosition;
23         this.playerName = playerName;
24         this.playerScore = playerScore;
25     }
26 
27     public String getPlayerPosition(){
28         return playerPosition;
29     }
30 
31     public String getPlayerName(){
32         return playerName;
33     }
34 
35     public String getPlayerScore(){
36         return playerScore;
37     }
38 
39 
40     public void setPlayerPosition(String playerPosition){
41         this.playerPosition = playerPosition;
42     }
43 
44     public void setPlayerName(String playerName){
45         this.playerName = playerName;
46     }
47 
48     public void setPlayerScore(String plaeyerScore){
49         this.playerScore = playerScore;
50     }
51 }
```

In the “Player” class, information such as position, name, and score are involved in this class. This class assigns different information with the correct input and output. This class makes sure that each value is corresponding together. For example, this program will return the player’s position when the position is requested. It can be assured that this program will not return the player's name when requesting for the position.

/Users/e2120564/NetBeansProjects/RugbyIA/src/Game.java

```
14 public class Game{
15     private String gameNumber = "not yet set";
16     private String gameAgainst = "not yet set";
17     private String gameResult = "not yet set";
18     private ArrayList<Player> playersChart = new ArrayList<Player>();
19 
20     public Game(){
21 
22     }
23 
24     public Game(String gameNumber, String gameAgainst, String gameResult, ArrayList<Player> playersChart){
25         this.gameNumber = gameNumber;
26         this.gameAgainst = gameAgainst;
27         this.gameResult = gameResult;
28         this.playersChart = playersChart;
29     }
30 
31     public String getGameNumber(){
32         return gameNumber;
33     }
34 
35     public String getGameAgainst(){
36         return gameAgainst;
37     }
38 
39     public String getGameResult(){
40         return gameResult;
41     }
42 
43     public ArrayList<Player> getPlayersChart(){
44         return playersChart;
45     }
46 
47      public void setGameNumber(String gameNumber){
48         this.gameNumber = gameNumber;
49     }
50 
51     public void setGameAgainst(String gameAgainst){
52         this.gameAgainst = gameAgainst;
53     }
54 
55     public void setGameResult(String gameResult){
56         this.gameResult = gameResult;
57     }
58 
59 
60 
61     public void setPlayerChart (ArrayList<Player> playersChart){
62         this.playersChart = playersChart;
63     }
64 
65 }
```

Data Structures Used

```
Game[] gamesArray = new Game[99];
private ArrayList<Player> playersChart;
private int rowCount = 0;
private int columnCount = 0;
private int counter = 0;
private ArrayList<Player> tempScores = new ArrayList<Player>();
```

I used arrays, 2d arrays, and ArraysList. This is considered an abstract data type, since 2d arrays are not a fundamental part of Java. The arrays were used to keep track of things like name list, score list, position, etc. The ArrayList was used to store a dynamically sized collection of elements. For example the Stat table uses ArrayList as the list has a dynamic size and dynamic data types.

Main Unique Algorithm

/Users/e2120564/NetBeansProjects/RugbyIA/src/NewIAIdea.java

```
389     private void addGameButtonActionPerformed(java.awt.event.ActionEvent evt) {
390         boolean a = gameNumberIN.getText().isEmpty();
391         boolean b = gameAgainstIN.getText().isEmpty();
392         boolean c = gameResultIN.getText().isEmpty();
393 
394 
395         if (ScoreTable.getRowCount() == 0) {
396             JOptionPane.showMessageDialog(null, "You haven't entered any users!");
397         } else {
398             if(!(a||b||c)) {
399                 if(ScoreTable.getSelectedRow() > 0) {
400                     playersChart = tempScores;
401                     tempScores = new ArrayList<Player>();
402                     String gameNumber = gameNumberIN.getText();
403                     String gameAgainst = gameAgainstIN.getText();
404                     String gameResult = gameResultIN.getText();
405                     ArrayList<Player> playersChart = new ArrayList<Player>();
406                     gamesArray[counter] = new Game(gameNumber, gameAgainst, gameResult, playersChart);
407 
408                     DefaultTableModel model = (DefaultTableModel)StatTable.getModel();
409                     int selectedIndex = ScoreTable.getSelectedRow();
410                     model.addRow(new Object[]{gameNumberIN.getText(), gameAgainstIN.getText(), gameResultIN.getText(),
411                         PositionSelection.getSelectedItem(), ScoreTable.getValueAt(selectedIndex, 1),
412                         ScoreTable.getValueAt(selectedIndex, 0), ScoreTable.getValueAt(selectedIndex, 2)});
413                     gameNumberIN.setText("");
414                     gameAgainstIN.setText("");
415                     gameResultIN.setText("");
416                 } else {
417                     JOptionPane.showMessageDialog(null, "Please select a user from the table!");
418                 }
419             } else {
420                 JOptionPane.showMessageDialog(null, "Do not leave any fields blank!");
421             }
422         }
423     }
```

This code represents the algorithm for the “Add game” button in the “Game entry” page. This button allows the user to move and store all of the necessary information in one table, which is located in the “Stat” page.

In this algorithm, it starts with an if statement which will determine through the table in the “Game entry” page. If there is no user selected on the table or if no user inputted in the table, then the program will project a pop-up window which will state “You haven’t entered any user!”. After there is a selected player, the user will fill in information about the game. When done, the user will click “ok” and the game information will be displayed on the “stat” page along with the corresponding player information that was selected.

Software Tools Used

Netbeans IDE was the main software tool I used. IDE stands for integrated development environment. This software offers functions such as text editor for typing in code, debugger, compiler, and spell checker. One of the most useful features was the GUI features (Graphical user interface) which allowed the common features to be used easily without any coding for features.

grade