This document is a software requirements specification for a Java soccer tournament simulation program called STS. STS enables its users to design tournaments containing a soccer pitch (or field) and a variety of teams. The user can define each team’s strategy by prescribing the roaming regions of each of its players. Users may elect a random, rather than a custom strategy, which instructs STS to determine an arbitrary strategy automatically. Once the pitch and teams are specified, STS runs a complete simulated soccer tournament. In the tournament each pair of teams plays against each other twice. At the conclusion of the tournament, STS can output the final scores of the soccer matches and as well as output a tournament standings table, which shows the relative performance of the teams.

The STS requirements are either functional or non-functional. Functional requirements explain the features of the software from an end-user’s perspective. Non-functional requirements are constraints imposed by the software that are unrelated to the software’s features.

1. Functional Requirements

1.1 Specify a soccer tournament

Before a soccer tournament can be played, the user must specify the soccer pitch upon which the tournament will be played, as well as specify a set of two or more teams that will participate in the tournament. The specifications must be provided to STS at execution time via a single file name argument. All specifications must be in a single text file. Following is a specification of the soccer pitch and the teams of a tournament.

1.1.1 Specify a soccer pitch

Defining its width and length attributes specifies a pitch. Both attributes are positive integers, whose values are in the range $1 \ldots (2^{31} - 1)$. The pitch is a Cartesian coordinate system with (0,0) as the bottom-left of the pitch and (maxWidth-1, maxLength-1) as the top-right of the rectangular pitch.
1.1.2 Specify a soccer team

1.1.2.1 Team name

A soccer team’s name is any number of words separated by whitespace characters. Names may consist of only alphabetic characters and whitespaces. I.e., a team’s name cannot have digits or special characters, such as hyphens.

1.1.2.2 Number of players on a team

A team may have any number of players in the range 1 .. \((2^{31}-1)\). However, it is customary for a team to have 11 players. Also, since each \((x,y)\) position of the pitch can be occupied by a single soccer player, team sizes should be selected so that all of the players from both the home and away teams can fit on the tournament pitch.

1.1.2.3 Team strategy

Each player of a team has a defined roaming range. That is a rectangular area of the pitch in which a player roams during the game. The only exception to a player being outside of his roaming range is kick off, where all players must retreat to their side of the pitch. Kick off occurs at the beginning of a game and after each goal is scored.

There is a maximum area for a player’s roaming rectangle region. For non-goalkeepers, the maximum area is the ratio of the total area of the pitch and the number of players on the team:

\[
\frac{\text{pitchWidth} \times \text{pitchLength}}{\text{teamSize}}
\]

The roaming region for goalkeepers is very specific (i.e., it is the pitch penalty box) and is identical to the roaming regions of all other goalkeepers of every team participating in the same tournament. The roaming region is defined by the rectangle coordinates \((x1, y1)\) and \((x4, y4)\), where the x and y values are defined as follows:

\[
x1 = \frac{\text{pitchWidth}}{6}
\]

\[
x4 = \text{pitchWidth} - x1
\]

\[
y1 = 0
\]

\[
y4 = y1 + \frac{\text{pitchLength}}{6}
\]

The roaming region of a goalkeeper cannot be modified once it has been established. Note that when a custom roaming region is specified via the STS input file, the first player of each team is the goalkeeper.
The STS program calculates the away roaming region automatically based on the home roaming region. In a tournament, the first team of each game is the home team and the second team is the away team. Since teams score on opposite ends of the pitch, STS calculates the away roaming region for each player as the mirror image of the home roaming region.

If a roaming region is not custom-specified for each player of a team in the STS input file, then STS assigns to each player of that team a random roaming region. This random region obeys the aforementioned restrictions on goalkeeper and player roaming regions area sizes. STS will terminate if it is provided with an input file that specifies ill-formed roaming regions (i.e., ones that exceed the aforementioned maximum) for any player in the tournament.

1.2 Play a tournament

An STS tournament involves a finite number of teams and a soccer pitch. Both are specified to the program via an input file. Once the teams and pitch are created and initialized, SSIM executes a tournament in which each team plays a game against each other team twice, once as a home team and once as an away team. Naturally, teams do not play against themselves. At the end of each game a score is produced, depending on the goals scored by each team during the game. The team with the most goals wins the game. If the two teams have the same number of goals in a game, the result is a tie. A team receives 3 points for a win, 1 point for a tie, and 0 points for a loss. The team with the most points wins the tournament.

Issuing the following command runs the STS program:

```
java -ea STS filename [s|st|ts|t]
```

The java program is invoked to run the JDK virtual machine upon which STS will execute. STS requires a file name to be specified as its first argument. An example file name is `attributes.txt`. The second argument is optional and can be one of the following:

- **s**: instructs STS to output the tournament scores between each pair of teams in a tabular format.
- **t**: instructs STS to output, in a tabular format, the tournament’s final standings table. This table shows how each team fared in the tournament and is sorted in non-ascending order based on the team’s earned points in the tournament.
- **st** or **ts**: instructs STS to output both the tournament scores and final standings table. Both of these options have the exact same output result.
### 1.2.1 Display tournament scores

At the conclusion of each tournament, the tournament scores will be displayed, provided that the user specified one of `s` or `st` or `ts` as an input argument to STS. An example tournament scores table is shown below:

<table>
<thead>
<tr>
<th>Manchester U</th>
<th>Chelsea</th>
<th>Liverpool</th>
<th>Arsenal</th>
<th>Manchester C</th>
<th>Aston Villa</th>
<th>Tottenham</th>
<th>Blackburn Rovers</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>2-1</td>
<td>3-0</td>
<td>2-0</td>
<td>7-0</td>
<td>2-1</td>
<td>0-0</td>
<td>3-1</td>
</tr>
<tr>
<td>1-2</td>
<td>---</td>
<td>3-0</td>
<td>4-0</td>
<td>5-0</td>
<td>2-0</td>
<td>2-1</td>
<td>4-1</td>
</tr>
<tr>
<td>0-5</td>
<td>0-3</td>
<td>---</td>
<td>0-2</td>
<td>0-3</td>
<td>0-1</td>
<td>0-2</td>
<td>0-5</td>
</tr>
<tr>
<td>0-5</td>
<td>0-5</td>
<td>1-0</td>
<td>---</td>
<td>4-1</td>
<td>1-3</td>
<td>2-2</td>
<td>2-8</td>
</tr>
<tr>
<td>0-9</td>
<td>0-15</td>
<td>0-1</td>
<td>0-3</td>
<td>---</td>
<td>0-0</td>
<td>0-2</td>
<td>3-5</td>
</tr>
<tr>
<td>1-4</td>
<td>0-1</td>
<td>0-0</td>
<td>0-2</td>
<td>0-0</td>
<td>---</td>
<td>1-0</td>
<td>1-2</td>
</tr>
<tr>
<td>1-3</td>
<td>0-2</td>
<td>0-0</td>
<td>0-1</td>
<td>2-2</td>
<td>0-0</td>
<td>---</td>
<td>0-3</td>
</tr>
<tr>
<td>2-6</td>
<td>4-8</td>
<td>2-0</td>
<td>4-1</td>
<td>14-0</td>
<td>4-1</td>
<td>8-1</td>
<td>---</td>
</tr>
</tbody>
</table>

Home results for each team are displayed horizontally and away results for each team are displayed vertically. Note that when reading scores, the score of the home team is displayed first. For example, Manchester United beat Liverpool twice in the tournament, 3-0 at home and 0-5 away. STS can truncate long team names in order to format the output scores table. For example, Manchester United is shown as Manchester U.

### 1.2.2 Display tournament standings

At the conclusion of each tournament, the tournament standings table will be displayed, provided that the user specified one of `t` or `st` or `ts` as an input argument to STS. An example tournament standings table is shown below:

<table>
<thead>
<tr>
<th>TEAM</th>
<th>HOME W</th>
<th>D</th>
<th>L</th>
<th>GS</th>
<th>GA</th>
<th>AWAY W</th>
<th>D</th>
<th>L</th>
<th>GS</th>
<th>GA</th>
<th>OVERALL P</th>
<th>D</th>
<th>L</th>
<th>GS</th>
<th>GA</th>
<th>GD</th>
<th>PTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manchester United</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>19</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>34</td>
<td>5</td>
<td>14</td>
<td>13</td>
<td>1</td>
<td>0</td>
<td>53</td>
<td>8</td>
<td>45</td>
</tr>
<tr>
<td>Chelsea</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>21</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>35</td>
<td>6</td>
<td>14</td>
<td>12</td>
<td>0</td>
<td>2</td>
<td>56</td>
<td>10</td>
<td>46</td>
</tr>
<tr>
<td>Blackburn Rovers</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>38</td>
<td>17</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>25</td>
<td>13</td>
<td>14</td>
<td>10</td>
<td>0</td>
<td>4</td>
<td>63</td>
<td>30</td>
<td>33</td>
</tr>
<tr>
<td>Arsenal</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>10</td>
<td>24</td>
<td>4</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>10</td>
<td>14</td>
<td>6</td>
<td>1</td>
<td>7</td>
<td>19</td>
<td>34</td>
<td>-15</td>
</tr>
<tr>
<td>Aston Villa</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>14</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td>18</td>
<td>-9</td>
</tr>
<tr>
<td>Tottenham</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>11</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>8</td>
<td>13</td>
<td>14</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>11</td>
<td>24</td>
<td>-13</td>
</tr>
<tr>
<td>Manchester City</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>35</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>32</td>
<td>14</td>
<td>1</td>
<td>3</td>
<td>10</td>
<td>9</td>
<td>67</td>
<td>-58</td>
</tr>
<tr>
<td>Liverpool</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>21</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>14</td>
<td>1</td>
<td>2</td>
<td>11</td>
<td>1</td>
<td>30</td>
<td>-29</td>
</tr>
</tbody>
</table>
The tournament standings table shows the team name (TEAM), its home record (HOME), its away record (AWAY), its overall record (OVERALL), its goal difference (GD), and its points earned (PTS). The home, away, and overall records are broken down into wins (W), ties or draws (D), losses (L), goals scored (GS), and goals accepted (GA).

2. Non-functional Requirements

2.1 Java constraints

JDK 1.6 or later must be used to run STS. Also, STS must be executed with the –ea option to enable runtime assertions, which are used throughout the STS source code.

2.1 STS Input file format

The contents of the STS input file contains the pitch and team attributes and has a specific format. An example file is shown below.

```xml
<begin pitch>
<length> 100 </length>
<width> 60 </width>
</end pitch>

<begin team>
<name> Manchester United </name>
<numberOfPlayers> 11 </numberOfPlayers>
<strategy> custom </strategy>
<region> (10,0) (50,16) </region>
<region> (0,11) (45,23) </region>
<region> (5,26) (59,36) </region>
<region> (6,77) (23,79) </region>
<region> (0,71) (5,79) </region>
<region> (11,54) (19,59) </region>
<region> (11,24) (19,29) </region>
<region> (42,7) (59,9) </region>
<region> (24,7) (41,9) </region>
<region> (13,76) (55,88) </region>
<region> (0,77) (56,86) </region>
```
<begin team>
  <name>Chelsea</name>
  <numberOfPlayers>11</numberOfPlayers>
  <strategy>random</strategy>
</end team>

<begin team>
  <name>Liverpool</name>
  <numberOfPlayers>11</numberOfPlayers>
  <strategy>random</strategy>
</end team>

<begin team>
  <name>Arsenal</name>
  <numberOfPlayers>11</numberOfPlayers>
  <strategy>random</strategy>
</end team>

<begin team>
  <name>Manchester City</name>
  <numberOfPlayers>11</numberOfPlayers>
  <strategy>random</strategy>
</end team>

<begin team>
  <name>Aston Villa</name>
  <numberOfPlayers>11</numberOfPlayers>
  <strategy>random</strategy>
</end team>
The BNF grammar of the input file is specified as follows:

```
RECORDS ::= {PITCH_RECORD | TEAM_RECORD}*
PITCH_RECORD ::= '<begin' 'pitch' '>' PITCH_ATTRIBUTES* '<end' 'pitch' '>
PITCH_ATTRIBUTES ::= PITCH_LENGTH | PITCH_WIDTH
PITCH_LENGTH ::= '<begin' 'length' '>' NUMBER '<end' 'length' '>
PITCH_WIDTH ::= '<begin' 'width' '>' NUMBER '<end' 'width' '>
TEAM_RECORD ::= '<begin' 'team' '>' TEAM_ATTRIBUTES* '<end' 'team' '>
TEAM_ATTRIBUTES ::= TEAM_NAME | TEAM_NUMBER_OF_PLAYERS | TEAM_STRATEGY
TEAM_NAME ::= WORD (' ' WORD)
TEAM_NUMBER_OF_PLAYERS ::= NUMBER
TEAM_STRATEGY ::= 'random' | 'custom' TEAM_REGIONS
TEAM_REGIONS ::= {'<region>' POINT POINT '<region>'}*
POINT ::= '{' NUMBER ',' NUMBER '}'
NUMBER ::= DIGIT NUMBER | DIGIT
DIGIT ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
WORD ::= LETTER WORD | LETTER
LETTER ::= 'a' | 'b' | ... | 'z' | 'A' | 'B' | ... | 'Z'
```