Software Requirement Specifications

for

STOCHASTIC CRICKET

Version 1.0

Team Members:

- Brijesh Kumar Singhal (Team Leader) – 145280011
- Kingshuk Pailan – 145280031
- Sumedh Gadpale – 145280006
- Kaustav Pakira – 145280026

Introduction

Here is a game which some say is dying or close to dying in its mother country England, in decline in the West Indies and unknown in most parts of the world.

But in India, cricket is "religion".

The country comes to a stop when a cricket match is being played - the roads are deserted, parties and weddings are postponed, operations in hospitals are rescheduled, parliament goes in for early closing.

In our project, we have tried to emulate this phenomenon by creating an extensive version of the hand-cricket game children play by pointing fingers in a random fashion.

A brief description

The desi version of this game consisted of two players randomly pointing fingers. One of them being the bowler and the other, batsman. The rule of scoring is very simple. If the number of fingers pointed match for both the players, the batsman gets out. If not, the number of fingers pointed by the batsman is considered his score for that particular ball.

In our project we allow the user to select if he/she wants to play a single-player or a two-player game.
In the single-player version, the user plays the game against the computer while in the two-player version two real-life players play and input the scores per ball in the computer. In both cases, we shall provide a statistical summary of the match.
In the single player version, the user shall be asked to specify a level of difficulty – amateur, semi-pro or pro corresponding to easy, medium or tough. The scoring system depends heavily on the level of difficulty. This is where probability comes into play.

The user will also have the option of choosing a particular team amongst a few pre-defined in-built teams. He/She can also create his/her own team and players and even modify one existing team.

We have two versions of play i.e 5 over and 10 over play. There will be scorecards and analysis available after each over or dismissal and at the innings' breaks.

So.. let's put on our pads and gloves. Let's have a swing of the bat and the bowl. Let's play.

A word of caution: Do not try to hit a six every ball. We believe that good bowlers should be respected. Play sensibly. Swing the bat for a six every time and its more probable that you might end up getting dismissed. This is just not cricket.. It's **STOCHASTIC CRICKET**.. !!

Gameplay

The game is modelled on the fact: what the batsman wishes to score on a particular ball and what the bowler wishes to achieve in that particular ball.

The user is first asked to select between 1-player or 2-player game.
In **1-player** game, he plays versus the computer. He is then asked to select a team for himself. He can achieve this by selecting one from a given set of teams or by creating his own. He will also be given the option to modify a team. Then he selects the opponent from the rest of the teams. After the teams are selected along with the players, there shall be a toss. Winner decides whether to bat or bowl first.
During the batting innings, user gets the opportunity to shuffle up the batting order according to his will. While batting, the objective of the user is to keep the computer guessing on how many runs he/she wants to score. If the guess of the computer matches, bad luck.
After each dismissal or at the end of an over, game statistics/analysis and match facts shall be provided.
During the bowling innings, user can select a bowler of his choice at the end of each over. Similar to batting the user now has to guess what the computer wants to score in that particular delivery. If he/she can guess correctly, check-mate.
There shall be commentary at the end of each ball. A detailed match summary and statistical inference shall be made at the end of the match.
In **2-player** version, the purpose of this game is simply database management. Two users play in real life and keep updating the scores to the computer. All functions and resources available for 1-player game will be available for 2-player game too.
The only difference in this format is that, initially, user has to select two teams before the match begins.

Method of Scoring

Let's see how the scoring works when this game is played manually.

Two players show a certain number of fingers at random. It is basically the number of runs he/she wants to score in that particular ball if he/she is batting or the number of runs he/she might concede in that particular ball while bowling. If the numbers match then the batsman is dismissed. If not, his scores increase by the number of fingers he pointed. The following table shows the scoring for a particular over.

Ball	1	2	3	4	5	6
Player - 1	2	1	4	6*	3	6*
Player - 2	3	6*	1	4	2	6*
Final Score	2	1	4	6*	3	out

Player-1 is batting while Player - 2 is bowling.
*For manual gameplay we assume the score 6 when a player points five fingers.
Thus the final score for the over is 2+1+4+6+3+out = 16.
If the scores of the players doesn't match, the score of the player batting is the runs scored in that particular ball. If it matches, the batsman is out.

In our game, the scoring method is based on similar lines, with a few modifications. The user has to input a number between one to six as his predicted score for that particular ball. In this manner, we shall have 7*7=49 such total ordered pairs(x,y), say, of scores with 'x' = score predicted by player1 and 'y'= score predicted by player2. Player2 is computer in case of 1-player game and user-2 in case of 2-player game. The ordered pairs are as follows:

Player 1 / Player 2	0	1	2	3	4	5	6
0	0,0	0,1	0,2	0,3	0,4	0,5	0,6
1	1,0	1,1	1,2	1,3	1,4	1,5	1,6
2	2,0	2,1	2,2	2,3	2,4	2,5	2,6
3	3,0	3,1	3,2	3,3	3,4	3,5	3,6
4	4,0	4,1	4,2	4,3	4,4	4,5	4,6
5	5,0	5,1	5,2	5,3	5,4	5,5	5,6
6	6,0	6,1	6,2	6,3	6,4	6,5	6,6

DISMISSALS

A batsman is dismissed when both the players predict the same score. So, there can be a total of 7 pairs when both scores are same.
So we assign the following dismissals:

Ordered Pair	Dismissal
(1,1)	LBW
(2,2)	LBW
(3,3)	Run Out
(4,4)	Bowled
(5,5)	Caught
(6,6)	Caught

We have kept two ordered pairs for extras – wide and no-ball.

Ordered Pair	Extra
(0,0)	Wide
(6,0)	No-Ball

The extras and the dismissals are same for all difficulty levels. The scoring depends on the difficulty level. The user shall input his own scores for each ball whilst the computer generates its own prediction using Monte Carlo Simulation where we attach specific weights or probability to the scores according to the difficulty level selected by the user.

AMATEUR :

Score	Probability
0	1/7
1	1/7
2	1/7
3	1/7
4	1/7
5	1/7
6	1/7
Total	1

The numbers generated between 0 and 6 by the computer shall be fully random where each number has equal probability of occurrence (=1/7).

SEMI-PRO:

Score	Probability
0	8/100
1	10/100
2	10/100
3	12/100
4	15/100
5	22/100
6	23/100
Total	1

The rationale behind assigning such probabilities of occurrence is simple – Greed. It is basic human nature to be naturally ambitious. So it is obvious that the user will go for big hits every ball. Thus, if the probability of generating a 4, 5 or 6 is high for the computer, it means that the more a user tries to hit a six, the more often he/she will end up being dismissed.
While the computer is batting, we keep the respective probabilities to be same. This is because the computer will now try to score more runs.

PRO:

Score	Probability
0	8/100
1	10/100
2	10/100
3	12/100
4	15/100
5	22/100
6	23/100
Total	1

We have kept the probabilities same as semi-pro but we have made a slight difference which makes batting a slight more difficult in this difficulty level.

If the user’s score and the computer generated score has an absolute difference of 1 given that both the scores lie within the interval [4,6], then by default it is going to be a dot ball with the subsequent match commentary saying that it was extremely close and the batsman must take precautions before going for a six next time.

Here, the rationale is same as in the semi-pro version, difference being in the fact that even if the computer guesses the user's score close to the original, if not exactly the original, it is going to be difficult to score. We have unanimously come to the decision that if the absolute difference is 1 for the higher numbers, then the batsman must settle for a dot-ball.

Similarly while the computer is batting, the weights to be assigned are same. Also the rule for the absolute difference for one stays.

Monte-Carlo Simulation (random data generation)

Monte Carlo methods (or **Monte Carlo experiments**) are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results; typically one runs simulations many times over in order to obtain the distribution of an unknown probabilistic entity. The name comes from the resemblance of the technique to the act of playing and recording results in a real gambling casino. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to obtain a closed-form expression, or unfeasible to apply a deterministic algorithm. Monte Carlo methods are mainly used in three distinct problem classes: optimization, numerical integration and generation of draws from a probability distribution.

Here we are concerned only with generation of draws from a probability distribution. Here we generate random integers from a given interval using the following in-built functions in c++. These functions belong to the stdlib.h and are as follows :

Function	Use
rand()	To generate random data
srand()	To generate random data from variable starting point
randomize()	This is used to randomise the starting point before generating the random number
random(n)	This generates a random number between 0 and n-1
RAND_MAX	The maximum value generated , the upper limit of the numbers to be generated.

The random numbers generated are to be divided by the value RAND_MAX and then convert the generated numbers to fractions. Thus we attach scores according to those fractions, depending on which range they lie in. The range we speak about here is the range in accordance with the probabilities attached earlier.

Others

We are going to use structures and classes for the scores statistics management and the teams management.