Deadline: 26th July, 2020

Graded Project Description

You need to design a console based **cricket game** and implement the following functionality.

- Bowling and batting must be continuously being called till one of the team wins the match (or in case of match draw)
- Whenever a bowling function is called, this will decrement the number of balls left.
- Whenever a batting function is called, this might (or might not) increment the score.
- Whenever a batsman is declared as ''out'', number of wickets of relevant team gets decremented
- In case of noball, update the values: +1 score, +1 ball
- In case of wide, +1 score

Some functions:

DecisionOfGame()
DecisionOfToss()
Batting()
Bowling()

There can be three categories of players: 1) Bowler, 2) Batsman, and 3) Allrounder.

To simulate **batsman behavior**, you will need to deal with bool value. True value indicates that the batsman has played the ball whereas false indicates that the batsman has acted defensive and skipped playing that ball.

Following four categories of balls must be considered:

- Noball
- Wide
- Bouncer
- Spin

To simulate the **behavior of bowling**, you need to keep following variables (a constant defined value and user entered value must be compared to label the category of ball). The attributes starting with const_ must be defined as a constant. You may hardcode any dummy value in it. For example, Const_distanceV = 50 feet

Const_distanceV: that tells how long is the pitch.
Entered_distanceV: value to be taken as an input from the user. This has to be compared with Const_distanceV to label the ball action as noball.
Const_distanceH: that tells the valid width allowed for bowling. (e.g. if the Entered_distanceH becomes larger than this value then it will result in a wide ball)
entered_distanceH: value to be taken as input from the user. This has to be compared with Const_distanceH to label the ball action as wide.
Const_height: maximum height allowed, above which the bowling will be considered as a bouncer.
Entered_height: value to be taken as input from the user. This has to be compared with Const_distanceV to label the ball action as a bouncer.
Entered_angle_of_deviation: value to be taken as input from the user. It tells the angle of deviation to decide whether the ball is a spin or not. Deviation of angle 5 degrees is allowed from 90 degrees. If deviation crosses +5 or -5 limit then this will be categorized as a spin.

Assuming the user won the toss and chose to bowl

Labelling rules: (No randomization allowed while labelling-compare user input with constant values and label)

Batsman behavior (Use randomization here, randomly assign value to isStrike flag)

Note: If the user chose to bat, then you have to perform reverse. You need to randomly assign values to Entered_distanceV and other relevant variables. The user will now tell whether to strike or not and you need to assign value to isStrike attribute using user input.

Batsman behavior

Ball label	**Criteria**	**If batsman strikes**	**If batsman does not strikes**
Noball	If Entered_distanceV < Const_distanceV	Choose randomly from 4, 6	0 runs
Wide	***add appropriate rule***	Always make 4 runs	0 runs
Bouncer	***add appropriate rule***	Always make 6 runs	0 runs
Spin	***add appropriate rule***	Choose randomly from 0,1,2,4, 6	0 runs

<u>Optional challenge</u>

Try changing the code that you have designed and add structs into your design. You can design structs for meaningful entities e.g. player.