Pseudocode Task - It's Just Not Cricket!

The variable table and the structured english algorithm describe a simplified version of a cricket match. A match consists of a user-specified number of overs. In this simplified version, one team bowls and one team bats. The aim is to achieve an average of more than 3 runs per over. At the end of each over the number of runs is recorded.

Identifier	Data Type	Purpose
NoOfOversInMatch	Integer	Stores the number of overs in the match (specified by the user)
NoOfOversPlayed	Integer	Stores the number of overs played so far
ScoreForThisOver	Integer	Stores the number of runs scored in this over
TeamScore	Integer	Stores the number of runs scored by the batting team
TargetScore	Integer	Stores the target score that a team must reach by the end of the match to win

```
TeamScore ← 0
TargetScore ← 0
OUTPUT "How many overs?"
INPUT NoOfOversInMatch
TargetScore = NoOfOversInMatch * 3
FOR NoOfOversPlayed ← 1 To NoOfOversInMatch Do
   OUTPUT "How many runs did the batting team score?"
   INPUT ScoreForThisOver
   TeamScore = TeamScore + ScoreForThisOver
ENDFOR
IF TeamScore >= TargetScore
   THEN OUTPUT "The batting team has won"
   ELSE OUTPUT "The bowling team has won"
ENDIF
```

Write a program for the above algorithm.

Test the program by showing the results of a match consisting of 4 overs with scores of 5 runs, 1 run, 0 runs and 2 runs.

Pseudocode Task - Rock, Paper, Scissors, Lizard, Spock

The variable table and the structured english algorithm describe a simplified version of a game called Rock, Paper, Scissors, Lizard, Spock. A match consists of a user-specified number of rounds. In this simplified version, the two players complete each game on paper and then enter information about the result of each game into a program that totals the number of games won by each player. Assume that all games have a winner ñ there are no drawn games.

Identifier	Data Type	Purpose
NoOfGamesInMatch	Integer	Stores the number of games in the match (specified by the user)
NoOfGamesPlayed	Integer	Stores the number of games played so far
PlayerOneScore	Integer	Stores the number of games won by Player One
PlayerTwoScore	Integer	Stores the number of games won by Player Two
PlayerOneWinsGame	Integer	Stores a 'Y' if Player One won the game and stores a 'N' otherwise

```
PlayerOneScore ← 0
PlayerTwoScore ← 0
OUTPUT "How many games?"
INPUT NoOfGamesInMatch
FOR NoOfGamesPlayed ← 1 TO NoOfGamesInMatch Do
   OUTPUT "Did Player One win the game (enter Y or N)?"
   INPUT PlayerOneWinsGame
   IF PlayerOneWinsGame = 'Y'
      THEN PlayerOneScore ← PlayerOneScore + 1
      ELSE PlayerTwoScore ← PlayerTwoScore + 1
   ENDIF
ENDFOR
OUTPUT PlayerOneScore
OUTPUT PlayerTwoScore
```

Write a program for the above algorithm.

Test the program by showing the results of a match consisting of three games where Player One wins the first game and Player Two wins the second and third games.