NCAA Tournament Simulator

VBA Final Project

Dave Allen

Contents

Executive Summary

As a college basketball connoisseur, I have noticed the lack of a sophisticated tool to predict the outcome of NCAA tournament games using the rankings from my favorite statistics website (kenpom.com).

The objective of this project is to provide an easy way to predict the outcomes of actual or hypothetical matchups between teams in the NCAA tournament. The project allows for an entire tournament simulation or a single game simulation. For each of these options, the results of each match is displayed, and also displayed is a summary of the decisive statistic is placed on a summary sheet in order to allow users to see why the outcome of the match happened the way it did.

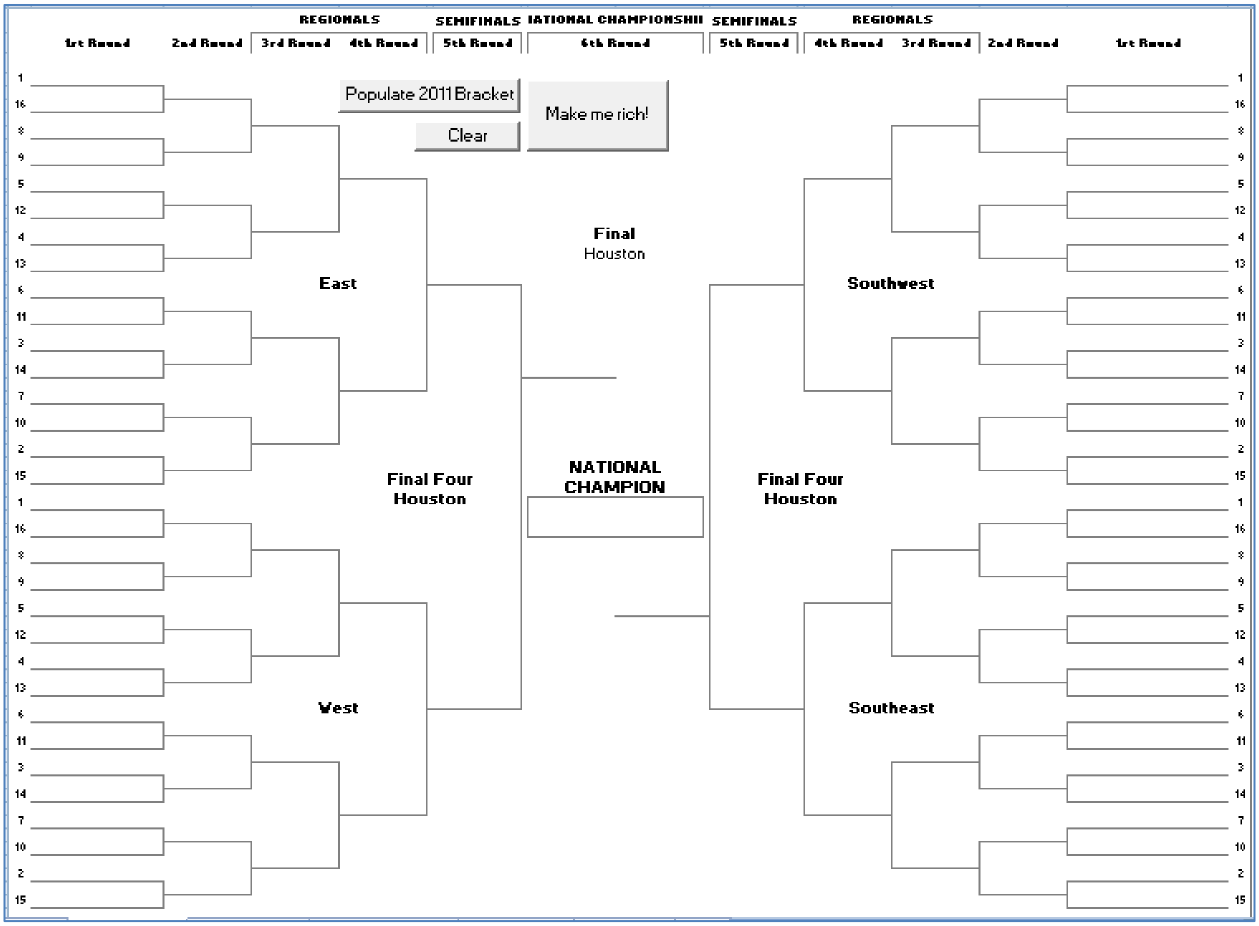
I feel like my project is a good solution to my original problem of finding a well-put-together tournament simulation, and hope that anyone else interested in college basketball would find my project informative and easy to use.

Implementation Documentation

The project has five worksheets on it, all with a separate function.

Bracket Sheet

The first and probably most used worksheet is the "Bracket" worksheet. This worksheet contains an empty bracket, as shown on the next page.

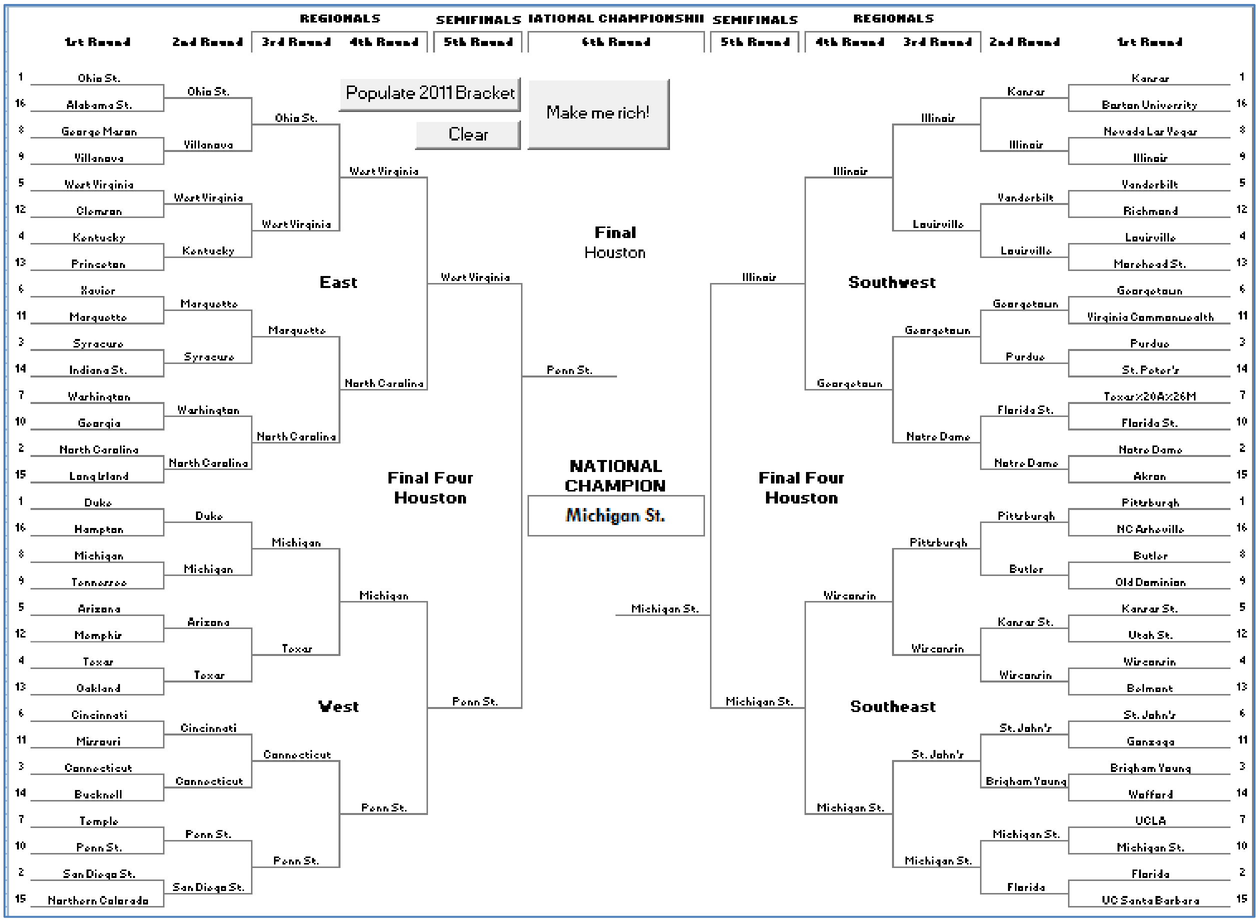


There are three buttons on this worksheet. The “Populate 2011 Bracket” button will populate the first round fields with the teams that were in the 2011 NCAA tournament. This makes it an easy way to simulate an entire tournament without having to enter in all of the 64 teams manually.

The “Clear” button does exactly what it seems like it would do. It clears the teams from every single round, essentially making the bracket ready to start fresh!

The last and most important button is the “Make me rich!” button. This button is the workhorse of the sheet. This button performs the actual tournament simulation. It takes each of the matchups in the first round, computes them, prints out the winner in the appropriate spot on the bracket, and then continues on to the next round until a winner is determined. The first-round teams can either be populated by the “Populate 2011 Bracket” button, or entered manually. If entered manually, the team names have to be entered exactly as they appear on the “Team List” worksheet, which will be covered next.

An example of a completed tournament after a successful “Make me Rich!” press is shown below.



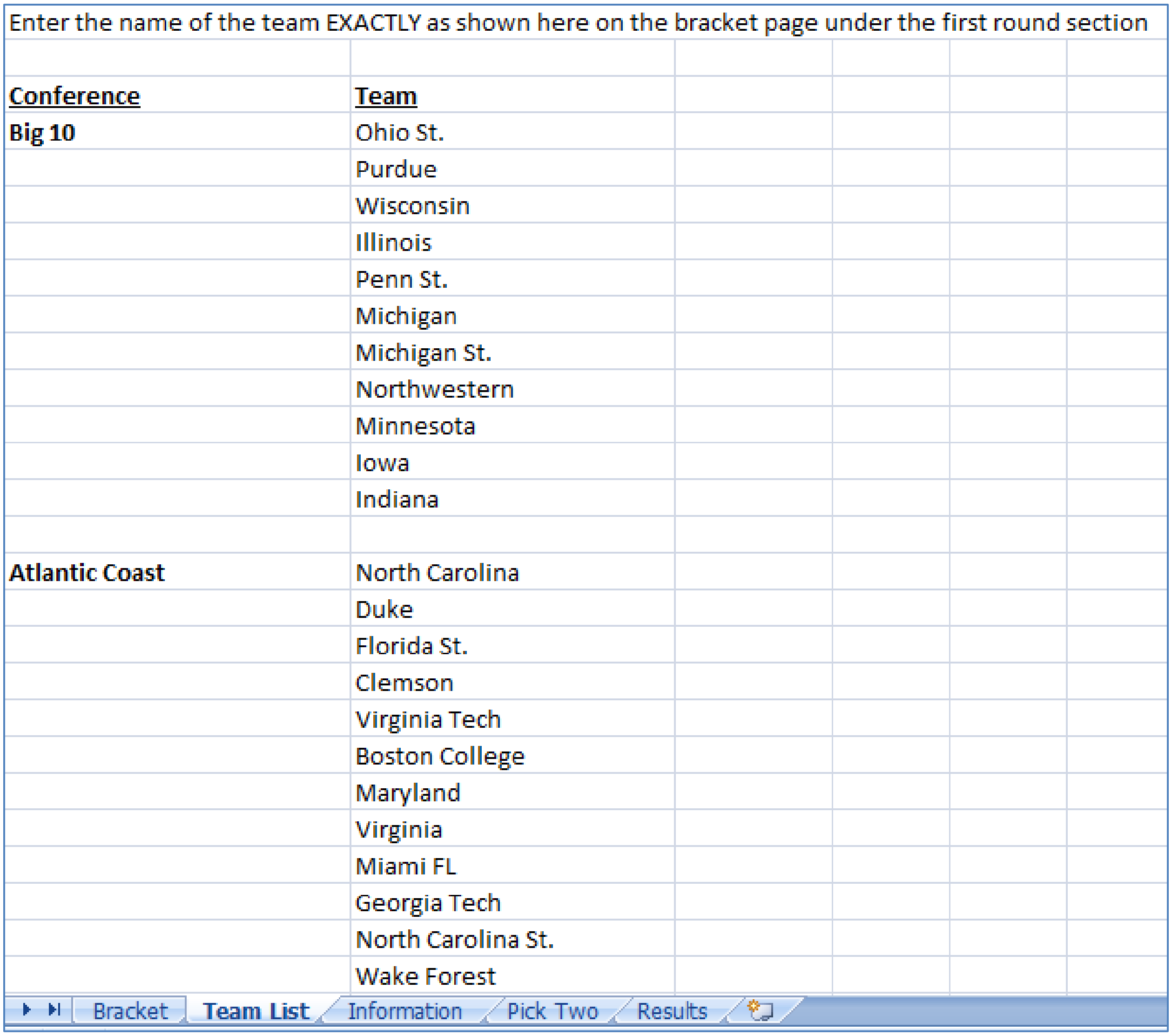
Seed	1st Round	2nd Round	3rd Round	4th Round	5th Round
1	Ohio St.	Ohio St.	Ohio St.	West Virginia	West Virginia
16	Alabama St.				
8	George Mason	Villanova			
9	Villanova				
5	West Virginia	West Virginia	West Virginia		
12	Clemson				
4	Kentucky	Kentucky			
13	Princeton				
6	Xavier	Marquette	Marquette	North Carolina	
11	Marquette				
3	Syracuse	Syracuse			
14	Indiana St.				
7	Washington	Washington	North Carolina		
10	Georgia				
2	North Carolina	North Carolina			
15	Long Island				
1	Duke	Duke	Michigan	Michigan	Penn St.
16	Hampton				
8	Michigan	Michigan			
9	Tennessee				
5	Arizona	Arizona	Texas		
12	Memphis				
4	Texas	Texas			
13	Oakland				
6	Cincinnati	Cincinnati	Connecticut	Penn St.	
11	Missouri				
3	Connecticut	Connecticut			
14	Bucknell				
7	Temple	Penn St.	Penn St.		
10	Penn St.				
2	San Diego St.	San Diego St.			
15	Northern Colorado				

6th Round	6th Round
Penn St.	Michigan St.

5th Round	4th Round	3rd Round	2nd Round	1st Round	Seed
Illinois	Illinois	Illinois	Kansas	Kansas	1
				Barton University	16
			Illinois	Nevada Las Vegas	8
				Illinois	9
		Louisville	Vanderbilt	Vanderbilt	5
				Richmond	12
			Louisville	Louisville	4
				Morehead St.	13
	Georgetown	Georgetown	Georgetown	Georgetown	6
				Virginia Commonwealth	11
			Purdue	Purdue	3
				St. Peter's	14
		Notre Dame	Florida St.	Texas%20A%26M	7
				Florida St.	10
			Notre Dame	Notre Dame	2
				Akron	15
Michigan St.	Wisconsin	Pittsburgh	Pittsburgh	Pittsburgh	1
				NC Asheville	16
			Butler	Butler	8
				Old Dominion	9
		Wisconsin	Kansas St.	Kansas St.	5
				Utah St.	12
			Wisconsin	Wisconsin	4
				Belmont	13
	Michigan St.	St. John's	St. John's	St. John's	6
				Gonzaga	11
			Brigham Young	Brigham Young	3
				Wofford	14
		Michigan St.	Michigan St.	UCLA	7
				Michigan St.	10
			Florida	Florida	2
				UC Santa Barbara	15

Team List Sheet

The team list sheet is simply a list of all the teams in Division I men's college basketball, sorted by conference. The team names listed here must be entered exactly as written onto the "Bracket" sheet in order for the algorithm to function properly. A snippet from the "Team List" sheet is shown below.



Enter the name of the team EXACTLY as shown here on the bracket page under the first round section

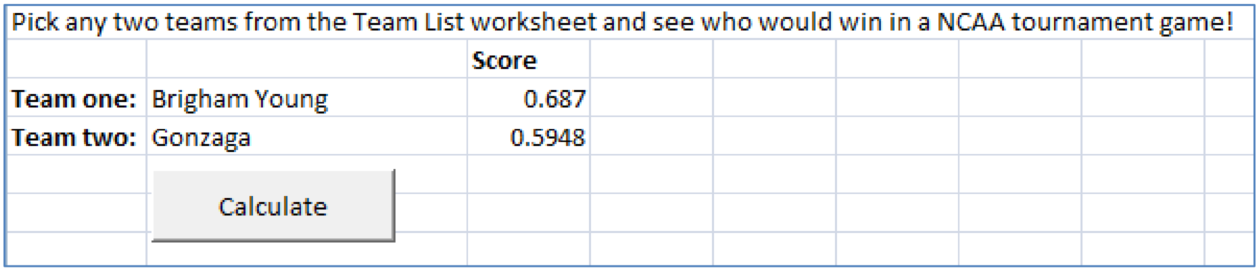
Conference	**Team**
Big 10	Ohio St.
	Purdue
	Wisconsin
	Illinois
	Penn St.
	Michigan
	Michigan St.
	Northwestern
	Minnesota
	Iowa
	Indiana
Atlantic Coast	North Carolina
	Duke
	Florida St.
	Clemson
	Virginia Tech
	Boston College
	Maryland
	Virginia
	Miami FL
	Georgia Tech
	North Carolina St.
	Wake Forest

Information Sheet

This sheet is simply used to hold data from the web scraping from the matchups. For each matchup, the Web Query Wizard scrapes Kenpom.com for the statistics for both teams. This information is pasted on this worksheet, and then the statistics are compared and winner is determined based on these stats. The Information sheet is normally hidden, and erases itself after every query.

Pick Two Sheet

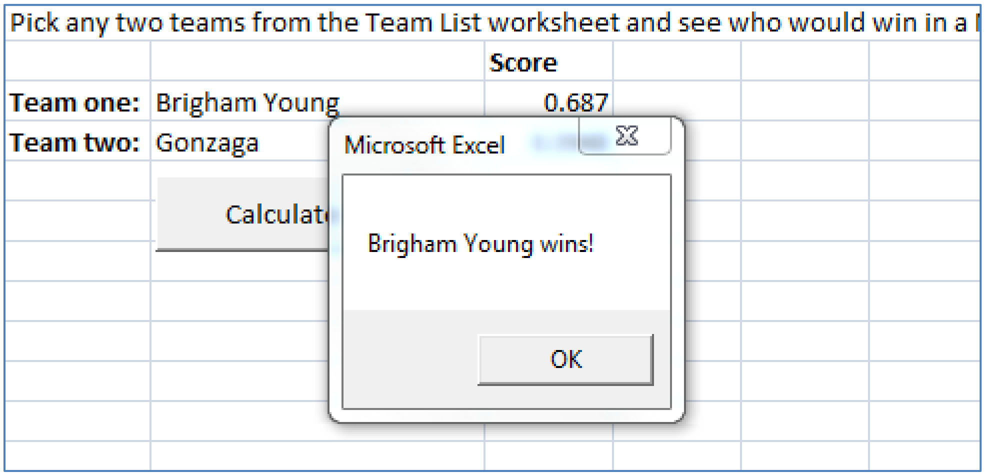
I created this sheet after realizing a need to figure out the outcome of one tournament game without simulating the whole tournament. The solution I came up with is a simple page where you can enter in two teams in the same format as the “Bracket” sheet.



Pick any two teams from the Team List worksheet and see who would win in a NCAA tournament game!

		Score
Team one:	Brigham Young	0.687
Team two:	Gonzaga	0.5948

After inputting the team names, simply press the "Calculate" button and watch the magic happen! The same algorithm used on the main "Bracket" sheet is applied here, and a message box will pop up with the winner. You can also see the scores that the algorithm assigned the two teams in the matchup after simulating a game.



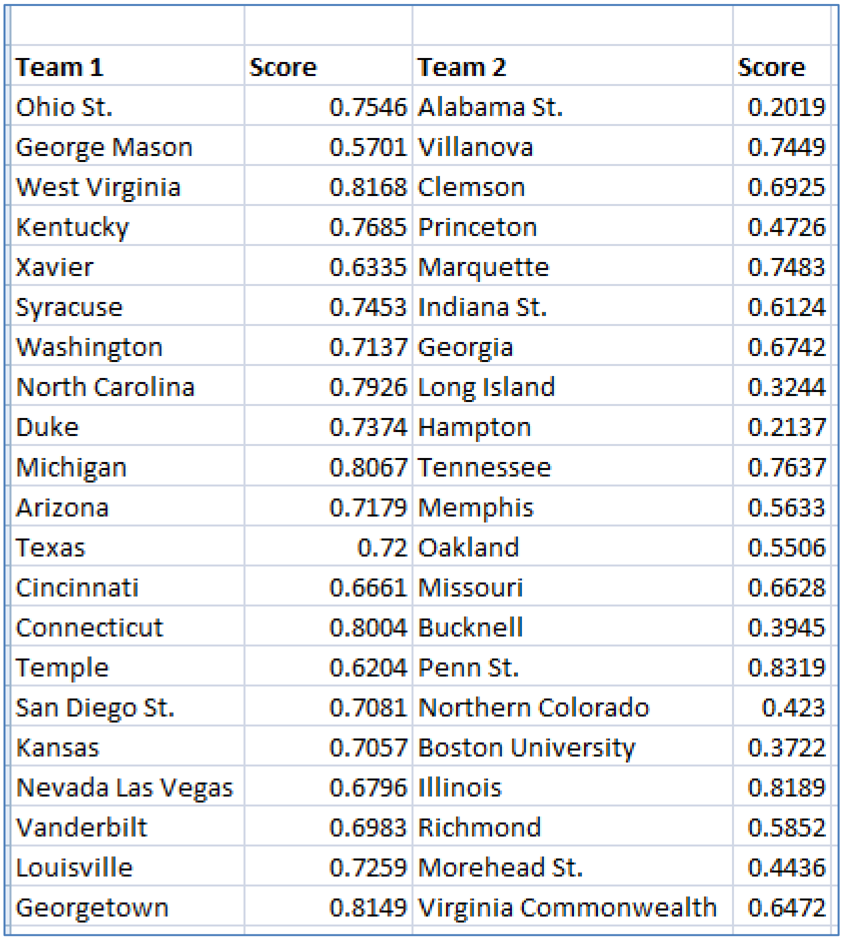
Pick any two teams from the Team List worksheet and see who would win in a

		Score
Team one:	Brigham Young	0.687
Team two:	Gonzaga	

Brigham Young wins!

Results Sheet

After first simulating a tournament, I realized that most users (myself included) would probably want to see the results of each of the matchups and see how close the scores of some of the teams in the matchups were. My solution to this problem consisted of adding a results page. After each tournament simulation, the results of each of the games are placed on this page for easy viewing as shown below.



Team 1	Score	Team 2	Score
Ohio St.	0.7546	Alabama St.	0.2019
George Mason	0.5701	Villanova	0.7449
West Virginia	0.8168	Clemson	0.6925
Kentucky	0.7685	Princeton	0.4726
Xavier	0.6335	Marquette	0.7483
Syracuse	0.7453	Indiana St.	0.6124
Washington	0.7137	Georgia	0.6742
North Carolina	0.7926	Long Island	0.3244
Duke	0.7374	Hampton	0.2137
Michigan	0.8067	Tennessee	0.7637
Arizona	0.7179	Memphis	0.5633
Texas	0.72	Oakland	0.5506
Cincinnati	0.6661	Missouri	0.6628
Connecticut	0.8004	Bucknell	0.3945
Temple	0.6204	Penn St.	0.8319
San Diego St.	0.7081	Northern Colorado	0.423
Kansas	0.7057	Boston University	0.3722
Nevada Las Vegas	0.6796	Illinois	0.8189
Vanderbilt	0.6983	Richmond	0.5852
Louisville	0.7259	Morehead St.	0.4436
Georgetown	0.8149	Virginia Commonwealth	0.6472

Learning and Conceptual Difficulties Encountered

The main difficulty I encountered with this project involved the Web Query Wizard. Getting all of the information I needed from it in a universal way was a bit of a challenge. One of the ways I got around this was to select the whole page instead of just a certain frame. I originally thought that the website had no ads because I have an ad-blocker on my main browser, so I figured that importing just the tables I wanted would work. Imagine my surprise when different data kept getting imported every time using the Web Query Wizard. This was easily solved once I diagnosed the problem, but sometimes diagnosing is the hardest part.

Another issue I encountered was just generally making the workbook user friendly. I'd like to be able to publish this somewhere and have other people be able to use it (although with a watered down algorithm so I don't give anything away!). To accomplish the user-friendliness, I tried to make things pretty self-explanatory, but then also having straight-forward instruction just in case it wasn't as straight forward as I would have hoped. Another thing I am planning to do is protect the worksheet, and lock all cells that users should not ever have a need to edit. I did not do this for the class project because I figured fellow students would like to look at the code, but for the potential mass distributed version, I will implement this locking and protection.

Another difficulty I encountered is how to choose the teams in the tournament. My first approach was a drop-down box, but due to the number of teams, the drop-down box was way too long and it took forever to find a team. My next approach was going to be a user form where you could select the conference, and then the team. This seemed like the best way to go about things, until I realized that doing this for 64 teams took way too long! My final solution was to allow users to simply type in the team name using the team list. I feel like this is the best approach because the way the teams have to be entered is pretty uniform, and once users get the feel for it they'll be able to enter in most teams without even looking at the team list, which would be much faster than both the drop-down box and the user form approaches.

The last difficult thing for me was simply the language. I have another class right now in VB.NET, and that combined with my little experience in VBA led to a little bit of frustration. I kept expecting some things that are really easy to do in VB.NET to be easy in VBA, but that definitely wasn't the case a lot of the time. This was solved simply by working at it. The more I coded, the more I learned and retained. Eventually I think I was able to develop a project that is well-coded and functions as its intended.