Brent Absher
MBA 614, Section 1

NCAA Tournament Analysis

Executive Summary:

I have participated in NCAA tournament bracket pools for several years. For the last five or six years, I have been using my own formula to rank the teams in the field as a starting point for my picks. With this strategy, I have won twice in those few years. This is not to say that I follow the quantitative analysis this project creates to the letter, but rather, it is a starting point. Other qualitative factors such as success in the last ten games, suspensions, etc. also play a part, but are considered on a case by case basis and are not factored in here (just as a company may be doing very well financially, but is being pummeled in the media, the numbers don't tell the whole story).

Although not strictly a "business problem," the analysis I created could be used in quite a number of financial analysis such as ratio, EPS, and other analyses. For example, if I were asked to analyze certain financial ratios of ten different companies within the same industry, without VBA automation (or another kind), I would need to pull their 10-K or other reports, plug the numbers into a spreadsheet, and churn the numbers from there. VBA gives the advantage in that I could automatically pull all these data from the respective websites and do the work for me. Although I could have done some kind of ratio analysis for my final project, I felt the NCAA bracket analysis would keep me more engaged in the process.

The system creates a data analysis based on my own formula. Please notice that there are two buttons on the NCAA Tourney tab on the ribbon to run the program. The first, designated by a picture of a basketball and named "Set Up," runs the entire program to calculate the raw score (although several programs from within module one are called). The second button opens the user form which interprets the data by comparing two selected teams. The advantage of this user form is that it is not dependent on the bracket (e.g., if you are doing a pool in which you play round by round rather than from the beginning, you can compare any two teams in the field even though they may have not played in the first round). By clicking on two teams within the list boxes, the program will evaluate whether a potential upset is plausible (or in the real world, how poorly one company may be doing in comparison to another). PLEASE note that the "Analyze Teams" button is dependent on the "Set Up" program being run first.

First, the program required gathering data from three different websites because more information was needed than any single website could give, including three separate URL's from one website. One of the more frustrating elements of the project is that, when data is pulled from three different websites, you have three different types of formatting (e.g. data every other line, data on every line, auto-formatting of dates, etc.). A MAJOR portion of this project was making the data jive with each other so (1) the data was readable, and (2) the data was useable.

Next, I needed to separate data within data. For example, one data point is the winning percentage against Top 50 RPI (ratings percentage index) opponents. However, the data doesn't show 9 wins out of 10 for a winning percentage of 90%. It shows a win/loss record of "9-1." This necessitated separating the digits so both the wins and win percentage could be calculated.

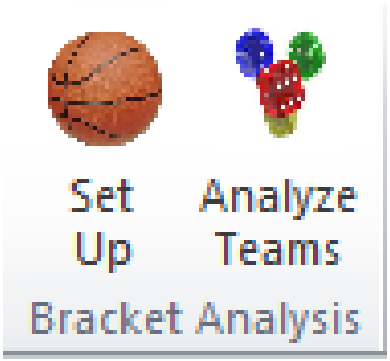
The third major portion of the project was the actual calculations of the raw data and the final raw score metric. This, in essence, was an education of loops on a massive scale. Every sheet depended on other sheets with a loop to have the correct data. This took a considerable amount of time to perform correctly.

Lastly, and most importantly since it is the end product, the user form as mentioned above evaluates the results. By clicking on two different teams, the form will evaluate the likelihood of an upset based on the raw score numbers.

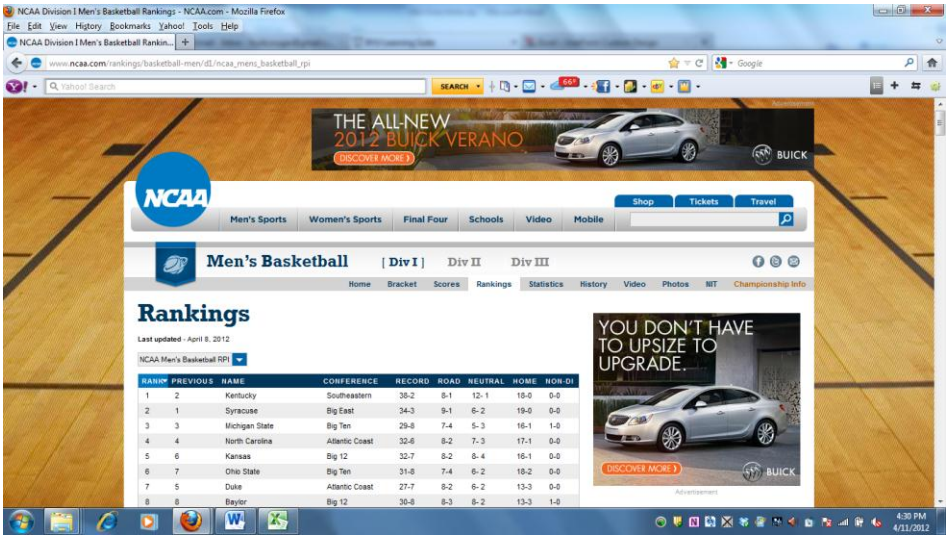
Implementation:

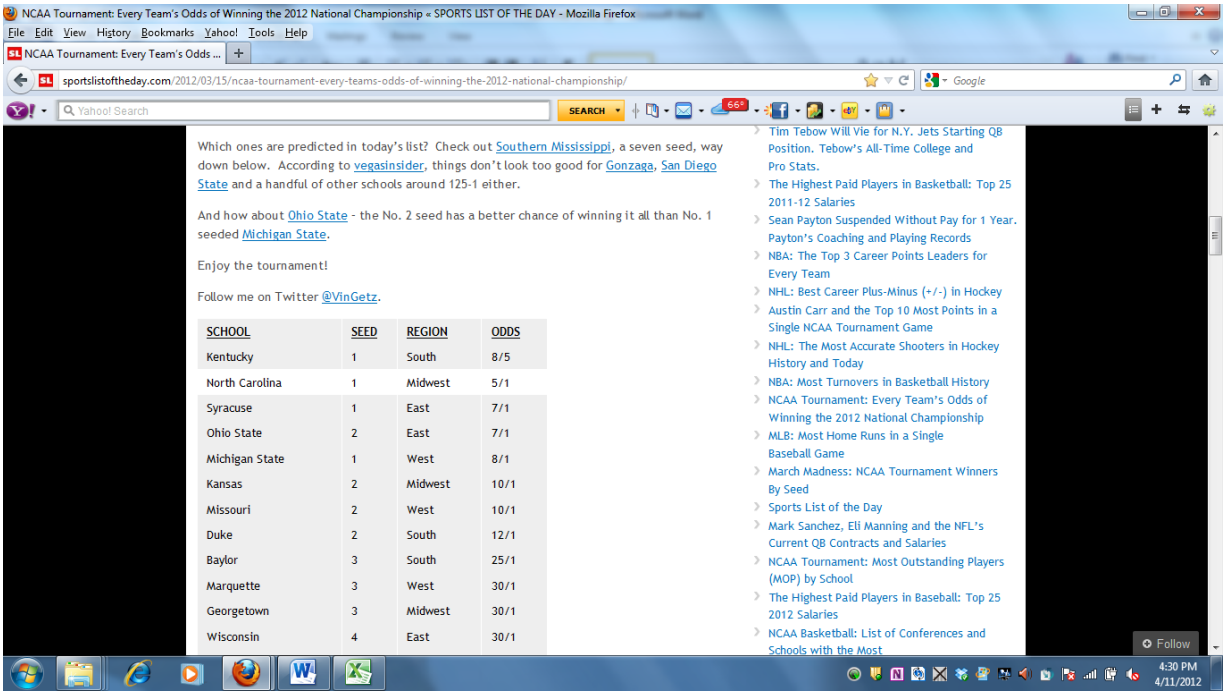
The project involved the implementation of the following:

1. Creating the two buttons below: the basketball sets up the spreadsheets and calculates the scores (please note that, because the data is coming from five separate web pages, this program takes 45 seconds to one minute to complete its function). The second button opens up the user form for the analysis once the calculation has completed.

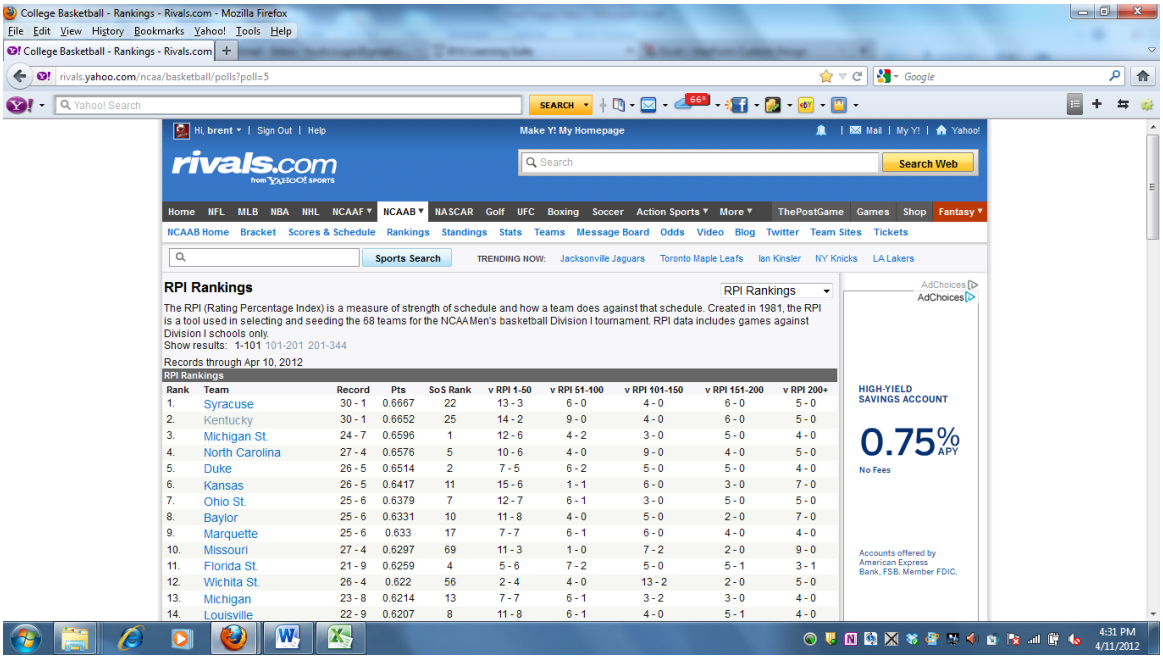


2. Pulling statistical data from the three separate websites. For one of these websites, three URL's needed to be used to pull the statistical data for all 344 division I NCAA basketball teams. Below are screen shots of the types of data I was dealing with. Auto-date formatting had to be turned off to make this data work correctly.

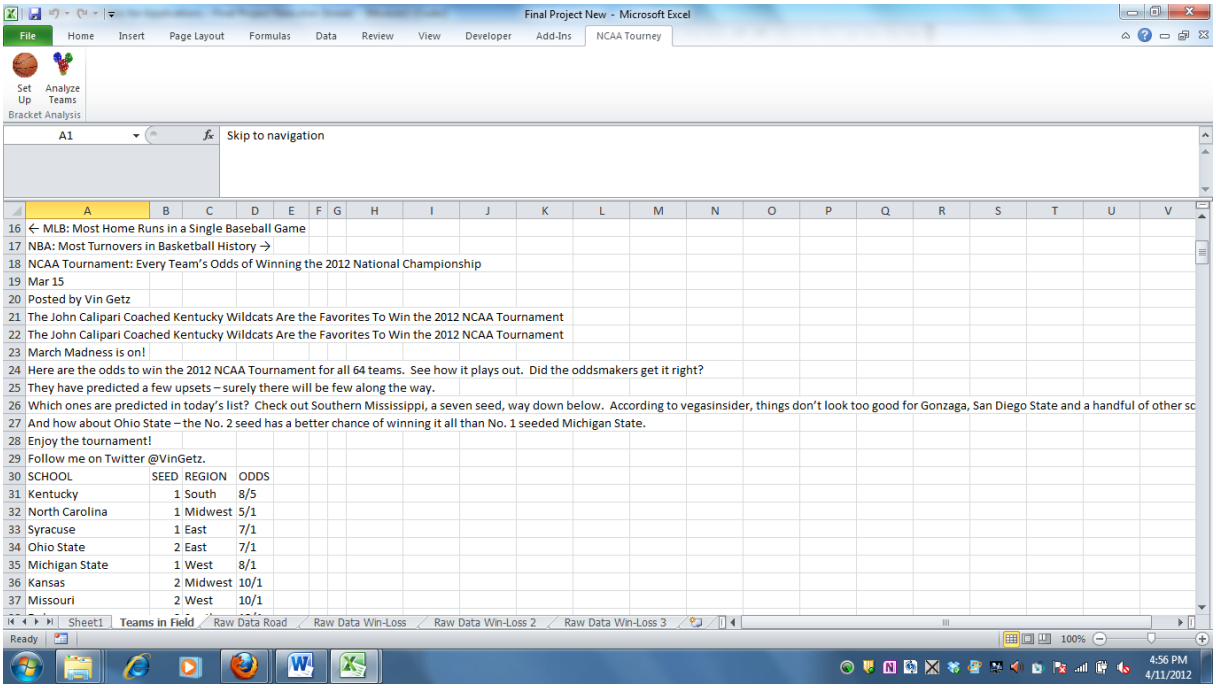




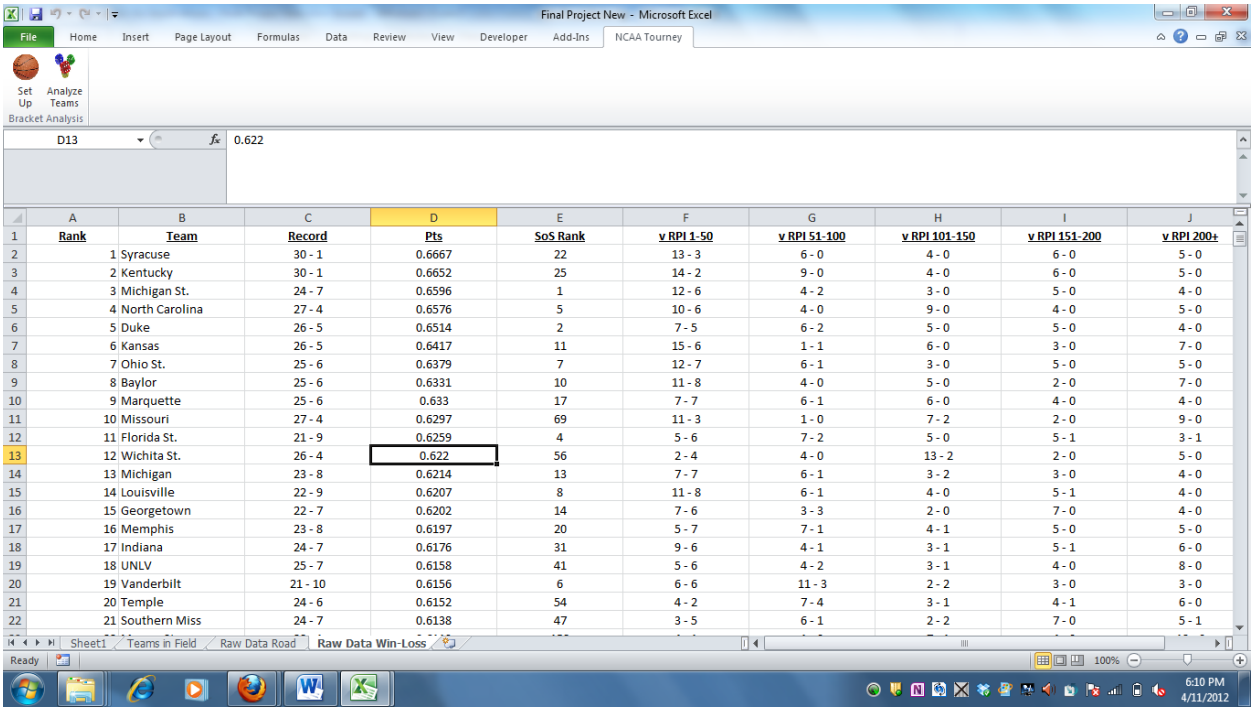
(This website required three URL's because only ~100 teams fit on a page. This is one of the pages)



3. Upon import, the data for each website was not very compatible with the data from the other sites. Below is a screen shot of the data from the first URL, containing hundreds of lines of useless data (the other URL's had additional information that was just as useless). Thus, a good portion of code was devoted to cleaning this up (you may note while running the program that screen updating is left on for the purpose of demonstrating just how much clean-up was needed).



Below is the screenshot of the data after being cleaned up:



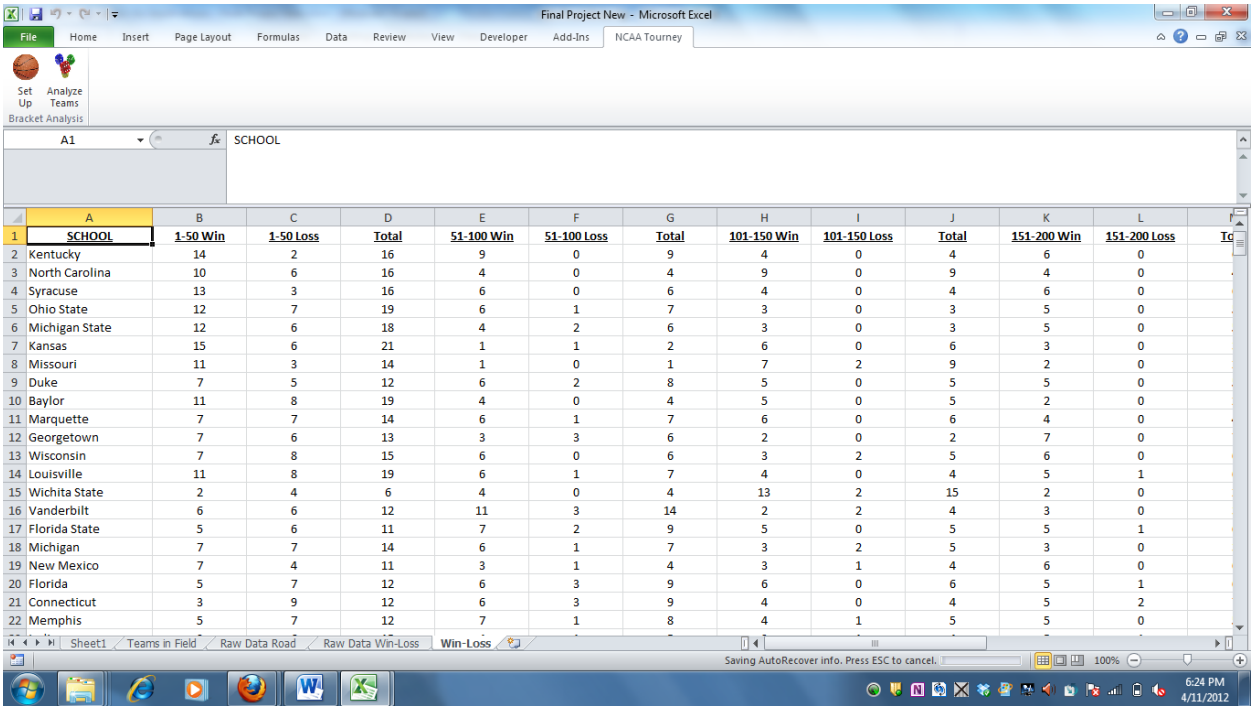
Rank	Team	Record	Pts	SoS Rank	v RPI 1-50	v RPI 51-100	v RPI 101-150	v RPI 151-200	v RPI 200+
1	Syracuse	30 - 1	0.6667	22	13 - 3	6 - 0	4 - 0	6 - 0	5 - 0
2	Kentucky	30 - 1	0.6652	25	14 - 2	9 - 0	4 - 0	6 - 0	5 - 0
3	Michigan St.	24 - 7	0.6596	1	12 - 6	4 - 2	3 - 0	5 - 0	4 - 0
4	North Carolina	27 - 4	0.6576	5	10 - 6	4 - 0	9 - 0	4 - 0	5 - 0
5	Duke	26 - 5	0.6514	2	7 - 5	6 - 2	5 - 0	5 - 0	4 - 0
6	Kansas	26 - 5	0.6417	11	15 - 6	1 - 1	6 - 0	3 - 0	7 - 0
7	Ohio St.	25 - 6	0.6379	7	12 - 7	6 - 1	3 - 0	5 - 0	5 - 0
8	Baylor	25 - 6	0.6331	10	11 - 8	4 - 0	5 - 0	2 - 0	7 - 0
9	Marquette	25 - 6	0.633	17	7 - 7	6 - 1	6 - 0	4 - 0	4 - 0
10	Missouri	27 - 4	0.6297	69	11 - 3	1 - 0	7 - 2	2 - 0	9 - 0
11	Florida St.	21 - 9	0.6259	4	5 - 6	7 - 2	5 - 0	5 - 1	3 - 1
12	Wichita St.	26 - 4	0.622	56	2 - 4	4 - 0	13 - 2	2 - 0	5 - 0
13	Michigan	23 - 8	0.6214	13	7 - 7	6 - 1	3 - 2	3 - 0	4 - 0
14	Louisville	22 - 9	0.6207	8	11 - 8	6 - 1	4 - 0	5 - 1	4 - 0
15	Georgetown	22 - 7	0.6202	14	7 - 6	3 - 3	2 - 0	7 - 0	4 - 0
16	Memphis	23 - 8	0.6197	20	5 - 7	7 - 1	4 - 1	5 - 0	5 - 0
17	Indiana	24 - 7	0.6176	31	9 - 6	4 - 1	3 - 1	5 - 1	6 - 0
18	UNLV	25 - 7	0.6158	41	5 - 6	4 - 2	3 - 1	4 - 0	8 - 0
19	Vanderbilt	21 - 10	0.6156	6	6 - 6	11 - 3	2 - 2	3 - 0	3 - 0
20	Temple	24 - 6	0.6152	54	4 - 2	7 - 4	3 - 1	4 - 1	6 - 0
21	Southern Miss	24 - 7	0.6138	47	3 - 5	6 - 1	2 - 2	7 - 0	5 - 1

4. The second major portion of the project involved separating win-loss records to be able to calculate the weights and winning percentages. As far as code is concerned, this was the most difficult portion of the project involving several complex loops. One of the major loops used to parse out this data is below:

   Loop #1:

```
Dim rowQ As Integer
Dim RoadWin As String

rowQ = 2

Do

RoadWin = WorksheetFunction.VLookup(Sheets("Win-Loss").Range("A" & rowQ), Sheets("Raw Data Road").Range("B1:H345"), 4, False)
Sheets("Win-Loss").Range("Q" & rowQ).Value = Left(RoadWin, InStr(1, RoadWin, "-") - 1)
rowQ = rowQ + 1

Loop Until Sheets("Win-Loss").Range("A" & rowQ) = ""
```

This results in the following workable spreadsheet:



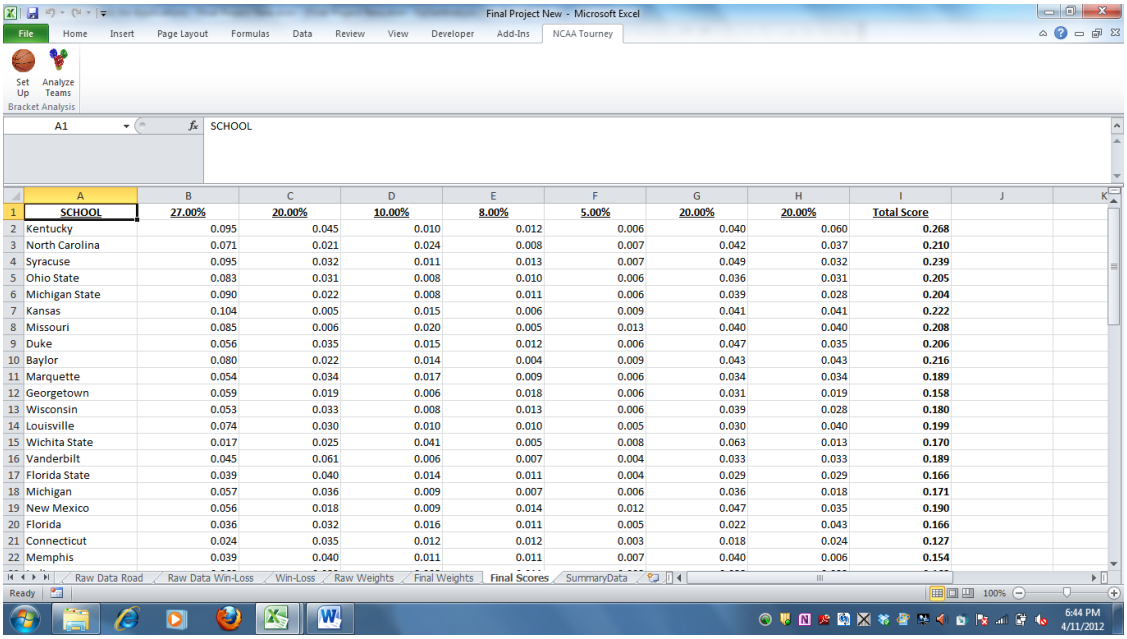
SCHOOL	1-50 Win	1-50 Loss	Total	51-100 Win	51-100 Loss	Total	101-150 Win	101-150 Loss	Total	151-200 Win	151-200 Loss
Kentucky	14	2	16	9	0	9	4	0	4	6	0
North Carolina	10	6	16	4	0	4	9	0	9	4	0
Syracuse	13	3	16	6	0	6	4	0	4	6	0
Ohio State	12	7	19	6	1	7	3	0	3	5	0
Michigan State	12	6	18	4	2	6	3	0	3	5	0
Kansas	15	6	21	1	1	2	6	0	6	3	0
Missouri	11	3	14	1	0	1	7	2	9	2	0
Duke	7	5	12	6	2	8	5	0	5	5	0
Baylor	11	8	19	4	0	4	5	0	5	2	0
Marquette	7	7	14	6	1	7	6	0	6	4	0
Georgetown	7	6	13	3	3	6	2	0	2	7	0
Wisconsin	7	8	15	6	0	6	3	2	5	6	0
Louisville	11	8	19	6	1	7	4	0	4	5	1
Wichita State	2	4	6	4	0	4	13	2	15	2	0
Vanderbilt	6	6	12	11	3	14	2	2	4	3	0
Florida State	5	6	11	7	2	9	5	0	5	5	1
Michigan	7	7	14	6	1	7	3	2	5	3	0
New Mexico	7	4	11	3	1	4	3	1	4	6	0
Florida	5	7	12	6	3	9	6	0	6	5	1
Connecticut	3	9	12	6	3	9	4	0	4	5	2
Memphis	5	7	12	7	1	8	4	1	5	5	0

5. Next, the weighted score must be calculated from the win-loss percentages using several loops. Note that when calculating these, a new sheet for each computation so that the user can see the raw data. The advantage of this is that the program is not dependent on the initial worksheets (Sheet1, 2, and 3). One of the major loops for the calculations is below:

Loop #2:

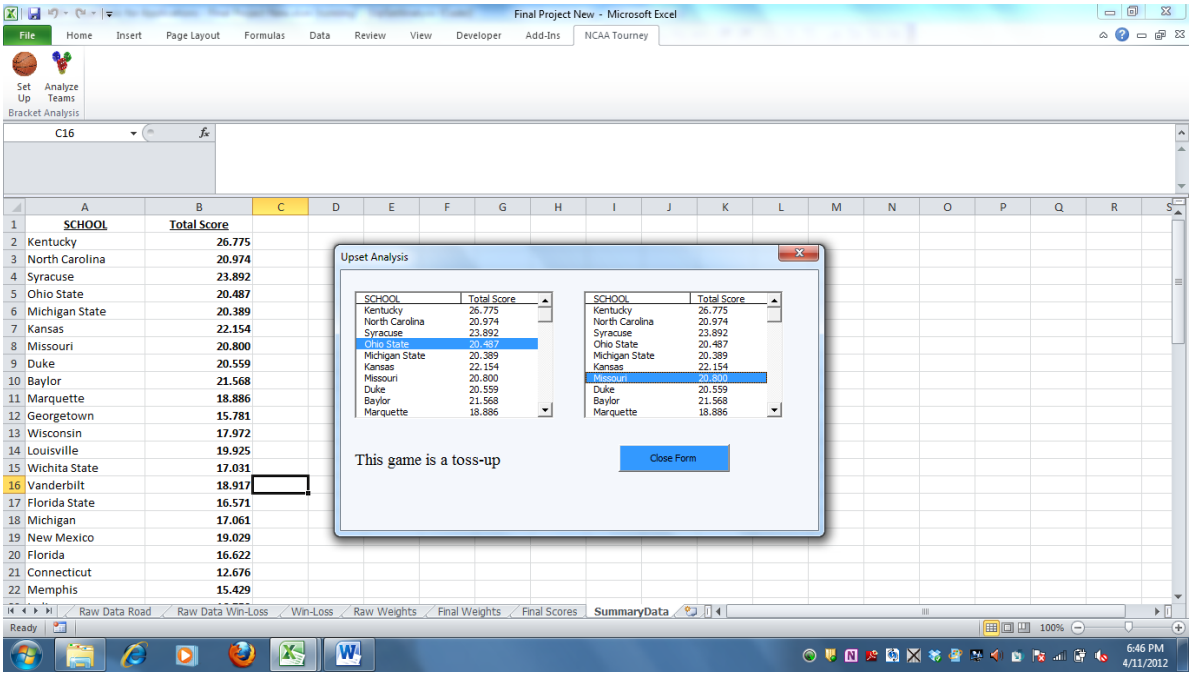
```
Dim LastRow As Integer
Dim CurrentRow As Integer
Dim Clmn As Integer

LastRow = Cells(Rows.Count, 1).End(xlUp).row

For Clmn = 4 To 22 Step 3

  For CurrentRow = 2 To LastRow

    Cells(CurrentRow, Clmn).Value = Cells(CurrentRow, Clmn - 2).Value + Cells(CurrentRow,
    Clmn - 1).Value

  Next

Next Clmn
```

6. After calculating the raw scores, each spreadsheet contains the relevant data for each step in the calculation, and of course, the final raw scores are in the last spreadsheet. The final product of the program is detailed below:



SCHOOL	27.00%	20.00%	10.00%	8.00%	5.00%	20.00%	20.00%	Total Score
Kentucky	0.095	0.045	0.010	0.012	0.006	0.040	0.060	**0.268**
North Carolina	0.071	0.021	0.024	0.008	0.007	0.042	0.037	**0.210**
Syracuse	0.095	0.032	0.011	0.013	0.007	0.049	0.032	**0.239**
Ohio State	0.083	0.031	0.008	0.010	0.006	0.036	0.031	**0.205**
Michigan State	0.090	0.022	0.008	0.011	0.006	0.039	0.028	**0.204**
Kansas	0.104	0.005	0.015	0.006	0.009	0.041	0.041	**0.222**
Missouri	0.085	0.006	0.020	0.005	0.013	0.040	0.040	**0.208**
Duke	0.056	0.035	0.015	0.012	0.006	0.047	0.035	**0.206**
Baylor	0.080	0.022	0.014	0.004	0.009	0.043	0.043	**0.216**
Marquette	0.054	0.034	0.017	0.009	0.006	0.034	0.034	**0.189**
Georgetown	0.059	0.019	0.006	0.018	0.006	0.031	0.019	**0.158**
Wisconsin	0.053	0.033	0.008	0.013	0.006	0.039	0.028	**0.180**
Louisville	0.074	0.030	0.010	0.010	0.005	0.030	0.040	**0.199**
Wichita State	0.017	0.025	0.041	0.005	0.008	0.063	0.013	**0.170**
Vanderbilt	0.045	0.061	0.006	0.007	0.004	0.033	0.033	**0.189**
Florida State	0.039	0.040	0.014	0.011	0.004	0.029	0.029	**0.166**
Michigan	0.057	0.036	0.009	0.007	0.006	0.036	0.018	**0.171**
New Mexico	0.056	0.018	0.009	0.014	0.012	0.047	0.035	**0.190**
Florida	0.036	0.032	0.016	0.011	0.005	0.022	0.043	**0.166**
Connecticut	0.024	0.035	0.012	0.012	0.003	0.018	0.024	**0.127**
Memphis	0.039	0.040	0.011	0.011	0.007	0.040	0.006	**0.154**

1. Finally, using the user form on the SummaryData sheet, one can now evaluate the teams based on their scores. The user form interface is shown below:



SCHOOL	Total Score
Kentucky	**26.775**
North Carolina	**20.974**
Syracuse	**23.892**
Ohio State	**20.487**
Michigan State	**20.389**
Kansas	**22.154**
Missouri	**20.800**
Duke	**20.559**
Baylor	**21.568**
Georgetown	**15.781**
Wisconsin	**17.972**
Louisville	**19.925**
Wichita State	**17.031**
Vanderbilt	**18.917**
Florida State	**16.571**
Michigan	**17.061**
New Mexico	**19.029**
Florida	**16.622**
Connecticut	**12.676**
Memphis	**15.429**

Learning and Conceptual Difficulties:

I did run into the following learning and conceptual difficulties:

- Formatting issues because of the data coming from three different websites. In particular, the program required a fairly complex VLOOKUP function within the code. I understand VLOOKUP very well within Excel, but the function was not working properly. After a short amount of time, I discovered that data from one source had extra spaces at the end of the team name. I was able to fix this using a simple TRIM formula; however, the solution took quite a bit of trouble-shooting and caused a substantial amount of frustration.

- Loops while calculating the raw team scores. For the first sheet of calculations ("Win Loss" sheet), I used "Do Until" loops, which resulted in quite a few loops. I knew a better way existed to do this, but I just couldn't seem to grasp it until the second set of calculations ("Weights," "Final Weights," and "Final Scores" sheets). I realized that a For-Next loop embedded within another For-Next loop was a much cleaner way of doing this. However, I didn't realize this until I was finished with the first spreadsheet. I would say that the one thing this project has really helped me to understand much better is how to do loops, as this was a very large portion of the work involved.

- The If-Then statement for the user form was perplexing at first, but I was able to do this correctly with a little help from a classmate (see next section).

- Lastly, the project raised an issue with being able to continue in the coming years. The data and code is sound; however, many names for schools can be written differently (e.g., Michigan State as opposed to Michigan St.). On the surface, this is very easy to fix with a simple "find and replace all" function. The tricky issue is that other schools are "St. Mary's" and "St. Bonaventure," so the order in which you replace them is paramount. I had to replace St. Mary's and others like it first so that when I replaced "St.," the program didn't change St. Mary's to "State Mary's." Another issue with St. Mary's is that, for some reason, I could not match it up so the VLOOKUP would work probably in its case. In the end, I had to simple find it on one sheet, copy it, select another sheet, find it again, and copy the original to make it work (only St. Mary's was like this for some reason) This makes me wonder how much work it would be to update this in subsequent years.

Assistance Received:

I received minor assistance from David Long on the first two difficulties mentioned above. In these cases, David was able to explain the concept and, from there, I was able to write this code on my own. In terms of time, I would estimate that David helped me for about twenty minutes on these issues (without writing any code for me).

On the user form above, David helped me for half an hour and walked me through the large If-Then statement that was required. I would estimate that I was able to do three quarters of this work on my own, but David did help me immensely to understand the concept.