**Tournament Tracker**

Goal  
Create a tournament tracker where the computer will display which teams need to play in a single-elimination style bracket. At the end, the winner should be identified. The model is the NCAA basketball tournament bracket for March Madness.

Initial Requirements

1. Track games played and their outcome (who won).
2. Multiple competitors play in the tournament.
3. Creates a tournament plan (who pays in what order).
4. Schedules games.
5. A single loss eliminates a player.
6. The last player standing is the winner.

Follow-up Questions and Answers

1. How many players will the tournament handle? Is it a fixed number or variable? The application should handle a variable number of players in the tournament. The program should be able to accept any number of players.
2. If a tournament has less than the full complement of players, how do we handle it? (Exponent of 2 – 2,4,8,16,32,64)? A tournament with less than the perfect number (a multiple of 2) should add in “byes”. Certain people selected at random get to skip the first round and act as if they won.
3. Should the order of play be random or ordered by input order? The order should be random.
4. Should we schedule the game or are they played whenever? The games should be played in whatever order and whenever the players want to play them.
5. If the games are scheduled, how does the system know when to schedule the game for? The games are not scheduled.
6. If the games are played whenever, can a game from the second round be played before the first round is complete? No. Each round should be fully completed before the next round is played.
7. Does the system need to store a score of some kind or just who won? Storing a simple score would be nice. Just a number for each player. That way, the tracker can be flexible enough to handle a checkers tournament (the winner would have a 1 and the loser a 0) or a basketball tournament.
8. What type of front-end system should this system have (form, webpage, app, etc.)? The system should be s desktop system for now, but down the road we might want to turn it into an app or a website.
9. Where will the data be stored? Ideally, the data should be stored in a Microsoft SQL database but please put in an option to store to a text file instead.
10. Will this system handle entry fees, prizes, or other payouts? Yes. The tournament should have the option of charging an entry fee. Prizes should also be an option, where the tournament administrator chooses how much money to award a variable number of places. The total cash amount should not exceed the income from the tournament. A percentage-based system would also be nice to specify. Could divide the percentages up in various ways.
11. What type of reporting is needed? A simple report specifying the outcome of the games per round as well as a report that specifies who won and how much they won. These can just be displayed on a form or they can be emailed to tournament competitors and the administrator.
12. Who can fill in the results of a game? Anyone using the application should be able to fill in the game scores.
13. Are there varying levels of access? (Admin, user w/ read only) No. The only method of varied access is if the competitors are not allowed into the application and instead, they do everything via email.
14. Should this system contact users about upcoming games? Yes, the system should email users that they are due to play in a round as well as who they are scheduled to play.
15. Is each player on their own or can teams use this tournament tracker? The tournament tracker should be able to handle the addition of other members. All members should be treated as equals in that they all get tournament emails. Teams should also be able to name their team. If there’s a team captain, every person will get an email.

Preliminary Design

**Structure**: Windows Forms application and Class Library

**Data**: SQL and/or Text File

**Users**: One at a time on one application

Key Concepts

Email – How do I email from C#

SQL

Database access in C#

Custom Events – Might be used to trigger actions, such as when to move to another round.

Error Handling – Prompt user to input correct data.

Interfaces – To pass information back and forth between forms

Random Ordering

(Bonus) Texting – Potentially add this feature even if not in requirements, because some users would prefer text notifications in addition to email

Data Mapping

Team

* Id (integer)
* teamMembers (List<Person>)
* teamName(string)

Person

* id (integer)
* firstName (string)
* lastName (string)
* emailAddress (string)
* cellPhoneNumber (string)

Tournament

* id (integer)
* tournamentName (string)
* contestantCount (int)
* entryFee (decimal)
* enteredTeams (list<team>)
* payoutType (string)
* prizes (list<prize>)
* rounds (list<list<matchup>>)
  + The first list = 1 round
  + The second list = matchups within the round

Prize

* plsceNumber (int)
* placeName (string)
* prizeAmount (decimal)
* pricePerentage (double)

Matchup

* entries (list<matchupEntry>)
* winner (team)
* matchupRound (int)

matchupEntry

* teamCompeting (team)
* score (double)
* parentMatchup (matchup)