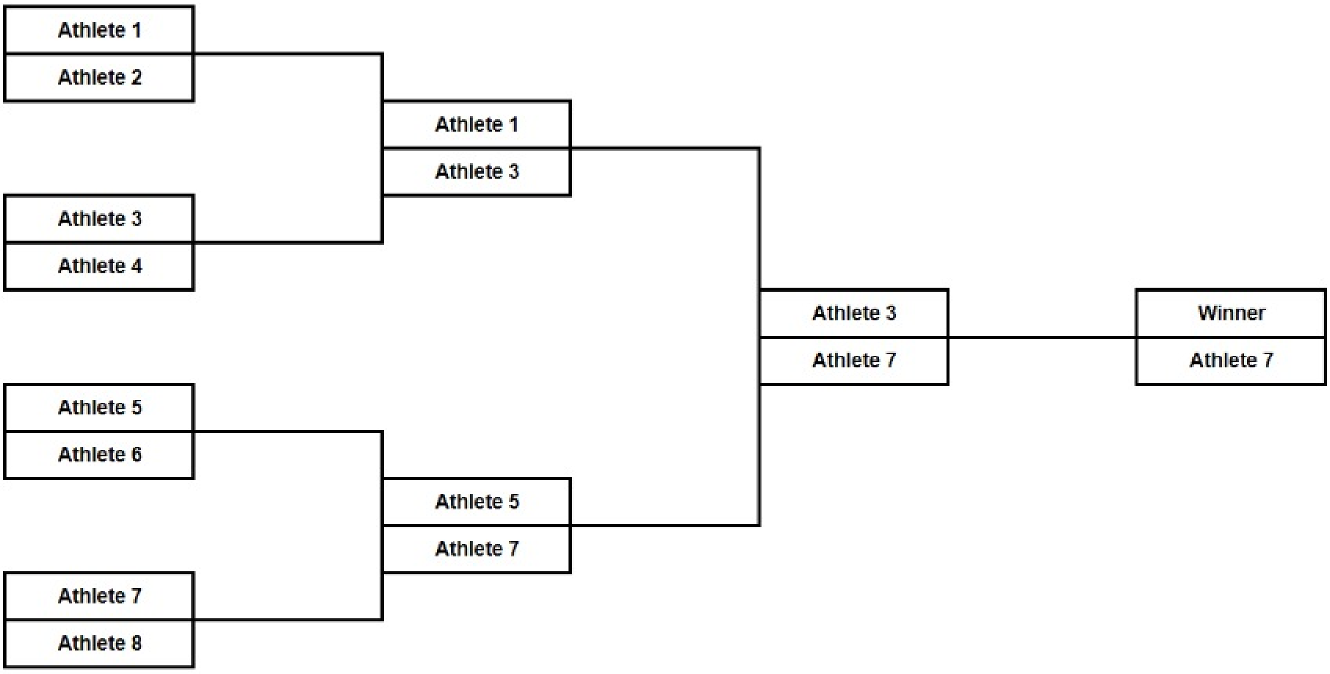
Lab 1

Implement an ADT to represent a tournament as show in the image.



- Assume the players are represented using integers from 1 to 2^x.
- Assume all the matches have been concluded before type 2 queries
- For a printing a match. assume you have to print the following:
    - `Match between p1 and p2, winner is pw`, where p1 and p2 are the players participating in the match (order does not matter), and pw is the winner among them
- All the input for the queries should be handeled in the `main.c` file. Other than that, there are no restrictions on the file structures.
- No need to write efficient/optimized code
- Assume no incorrect input will be given so no need to check input

You have to implement the following functionalities for the tournament:
Type 1:

- create_tournament n -> Creates a tournament with n participants.
    - Will be the first query.
    - Will only be given once
    - n will be a power of 2
- create_match p1 p2 pw -> Creates a match between p1 and p2 with the winner as pw

  - It is guaranteed while providing the input that if a match between p1 and p2 is happening and it is not their first match, then all the previous matches for p1 and p2 have already been provided

- end_matches -> Signals the end of queries of type 1

Type 2:

- print_tournament -> Should print out all matches, order does not matter.
- print_stages -> Should print out all matches in stages. The stages in the above example are:
  - Stage 0: (1, 2), (3, 4), (5, 6), (7, 8)
  - Stage 1: (1, 3), (5, 7)
  - Stage 2: (3, 7)
  - No need to print stage 0, stage 1, etc. Just print the matches in order of Stage 0, Stage 1, Stage 2, ...
  - Order of matches within a stage does not matter
  - You can implement it so that the output from `print_tournament` and `print_stages` is the same
- print_history p1 -> Should print the history of matches of p1 in order
  - Eg: `print_history 5` should print the matches between 5 and 6, and 5 and 7 in that order
  - Eg: `print_history 3` should print the matches between 3 and 4, 3 and 1, and 3 and 7 in that order
- end -> Queries are over, exit the program