Siddhant Achrekar

CS460-01

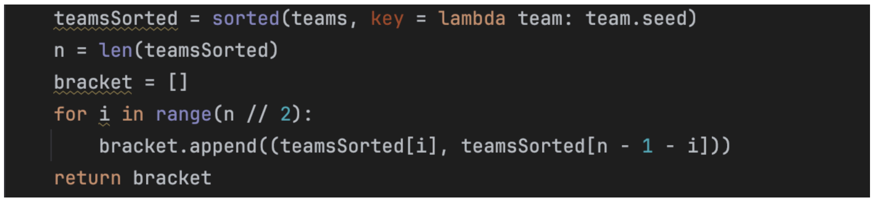
Dominic Dabish

July 3, 2025

Final Project Module 3

1. Creating a fair and visually intuitive tournament bracket is critical in sports like soccer, basketball, football, and so on. This project implements a seed-based bracket pairing algorithm that ensures fair and balanced matchups. It's designed for organizers of single-elimination tournaments (UEFA Champions League or NBA Playoffs), ensuring that higher seeds are appropriately rewarded with the structure being easy to understand.
2. Core Algorithm Description and Pseudocode

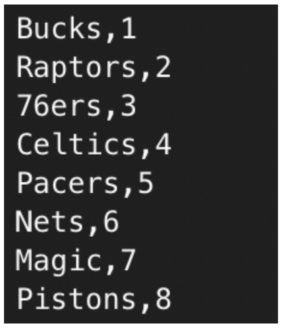
- The algorithm sorts teams and matchups by seed.
- It pairs the lowest with the lowest (1 v 16, 2 v 15, …)
- Visualizes the bracket matchups.



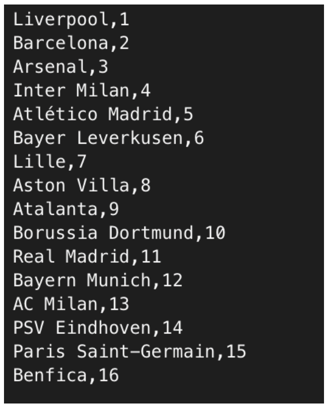
```
teamsSorted = sorted(teams, key = lambda team: team.seed)
n = len(teamsSorted)
bracket = []
for i in range(n // 2):
    bracket.append((teamsSorted[i], teamsSorted[n - 1 - i]))
return bracket
```

3. Test Plan and Results.

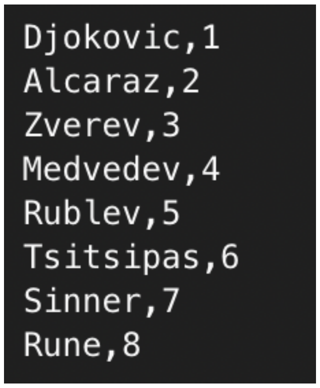
Input Test Cases:



```
Bucks,1
Raptors,2
76ers,3
Celtics,4
Pacers,5
Nets,6
Magic,7
Pistons,8
```

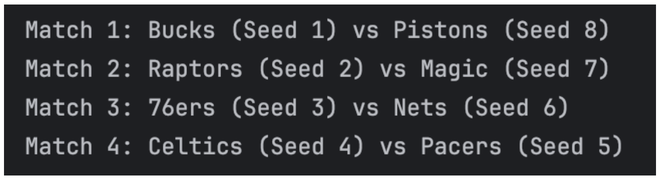


```
Liverpool,1
Barcelona,2
Arsenal,3
Inter Milan,4
Atlético Madrid,5
Bayer Leverkusen,6
Lille,7
Aston Villa,8
Atalanta,9
Borussia Dortmund,10
Real Madrid,11
Bayern Munich,12
AC Milan,13
PSV Eindhoven,14
Paris Saint-Germain,15
Benfica,16
```

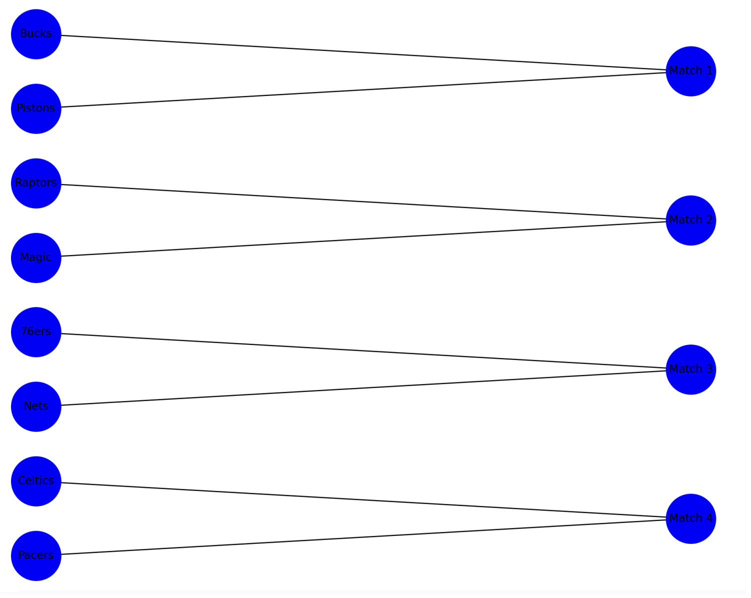


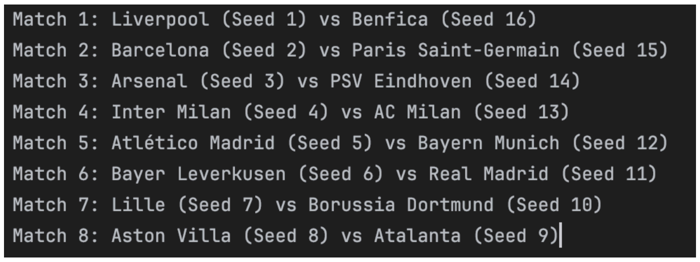
```
Djokovic,1
Alcaraz,2
Zverev,3
Medvedev,4
Rublev,5
Tsitsipas,6
Sinner,7
Rune,8
```

Expected:

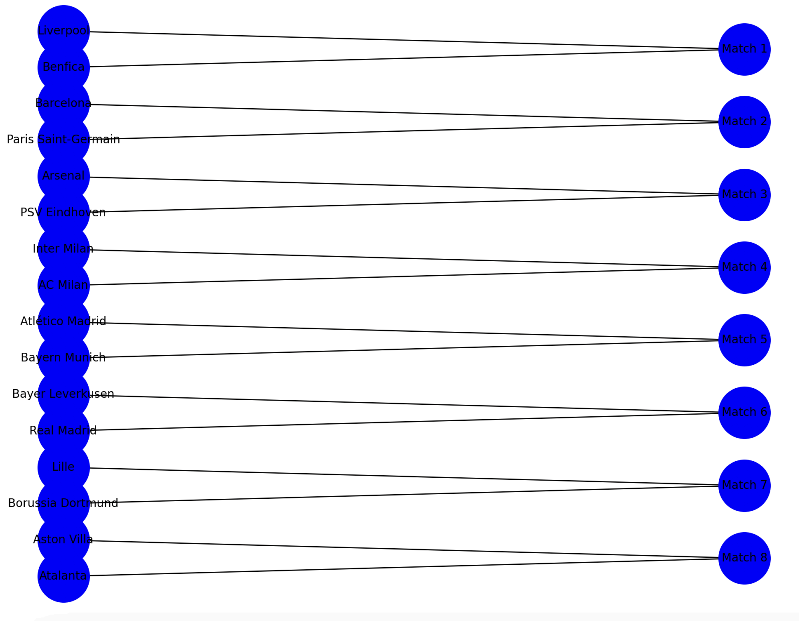


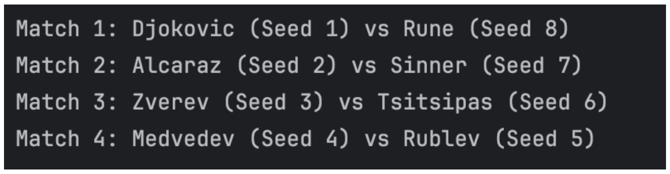
```
Match 1: Bucks (Seed 1) vs Pistons (Seed 8)
Match 2: Raptors (Seed 2) vs Magic (Seed 7)
Match 3: 76ers (Seed 3) vs Nets (Seed 6)
Match 4: Celtics (Seed 4) vs Pacers (Seed 5)
```

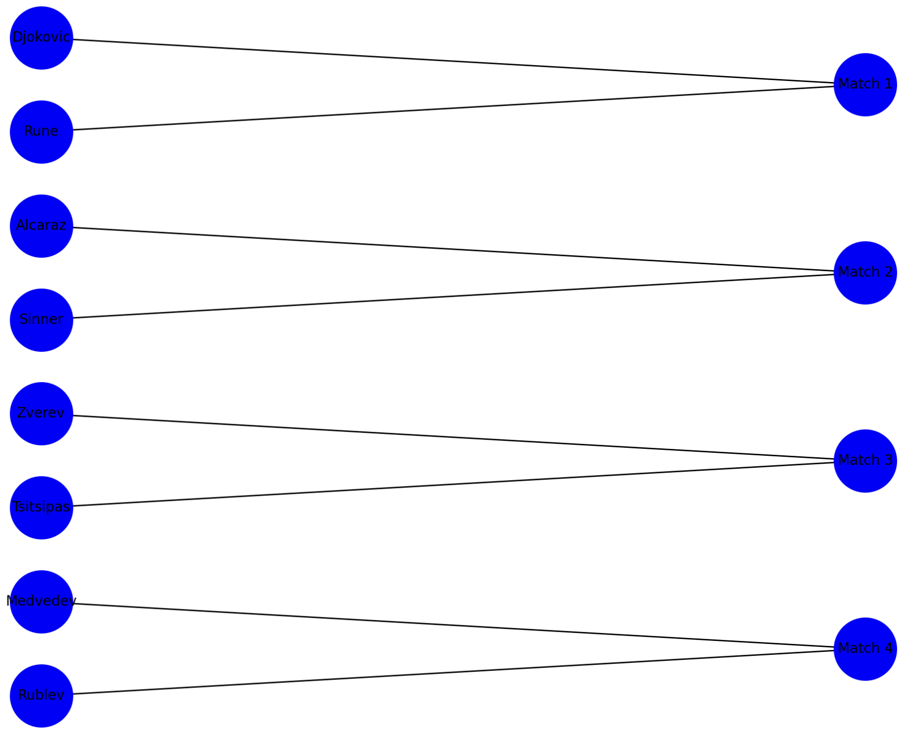




```
Match 1: Liverpool (Seed 1) vs Benfica (Seed 16)
Match 2: Barcelona (Seed 2) vs Paris Saint-Germain (Seed 15)
Match 3: Arsenal (Seed 3) vs PSV Eindhoven (Seed 14)
Match 4: Inter Milan (Seed 4) vs AC Milan (Seed 13)
Match 5: Atlético Madrid (Seed 5) vs Bayern Munich (Seed 12)
Match 6: Bayer Leverkusen (Seed 6) vs Real Madrid (Seed 11)
Match 7: Lille (Seed 7) vs Borussia Dortmund (Seed 10)
Match 8: Aston Villa (Seed 8) vs Atalanta (Seed 9)
```




```
Match 1: Djokovic (Seed 1) vs Rune (Seed 8)
Match 2: Alcaraz (Seed 2) vs Sinner (Seed 7)
Match 3: Zverev (Seed 3) vs Tsitsipas (Seed 6)
Match 4: Medvedev (Seed 4) vs Rublev (Seed 5)
```



4. Runtime / Memory Measurements
    i. Sorting Teams by Seed: O(n log n)
    ii. Pairing Logic: O(n)
    iii. Memory Usage: O(n)
    iv. Tools: time.time() for timing, matplotlib for rendering

5. Trade-Offs, Limitations, and Future Work
    - The current implementation assumes that the input format is perfect and an even number of teams. It does not simulate the winners or the next-round bracket. While it does visualize the bracket clearly, the layout is static. In the future, I plan to:
        - Add simulation of winners to advance to later rounds along with the tournament.
        - Save the visual output to a PDF/Image
        - Expand to handle larger amounts of teams (32, 64, etc.)
        - Expand to handle an odd number of teams.

6.

- https://github.com/sidachr/cs460Final