Main:

System Operations Explanation

1. Store the preferences for a Candidate
    - Each **Candidate** has an attribute **preferences: list[int]**, which is a list of company IDs sorted by preference.
    - This preference list is initialized when a **Candidate** object is created
2. Determine if every **Candidate** is match with their favorite company
    - The *is_perfect_matching(matching: List[Tuple[int, int]]) -> bool* method in **ListOfCandidates** iterates over the given matching list to get **candidate_id** and **company_id**.
    - Based on **candidate_id**, it gets **Candidate** from **list[Candidate]**. Then it retrieves each candidate's top preference using *get_top_choice()* in the **Candidate** class and checks if they are assigned to it (compared with the **company_id** just get).
3. Determine if every **Candidate** is matched with some **Company**

    The *is_complete_matching(matching: List[Tuple[int, int]]) -> bool* method in **ListOfCandidates** get the list of **matching_candidate_id** from the ***matching***

    Then it check if all candidates in the candidates list are matched (compared the length of **matching_candidate_id** and the length of **candidates: list[Candidate]**)
4. Determine if every **Candidate** is matched with a **Company** in their **N** top choice

    The *is_top_N_matching(matching: List[Tuple[int, int]], N: int) -> bool* method in **ListOfCandidates** loops through the **candidate_id** and **company_id** in **matching**.

    Then, it calls *is_top_N_choice(company_id: int, N: int) -> bool* in **Candidate** on each candidate to verify that their assigned company is within their top **N** choices.
5. Interaction between class

    Candidate handles individual candidate preferences

    ListOfCandidates manages multiple candidates and performs operations requiring multiple candidates (make calls to objects of type Candidate)