**Below are some major differences between Greedy method and Dynamic programming:**

| **FEATURE** | **GREEDY METHOD** | **DYNAMIC PROGRAMMING** |
| --- | --- | --- |
| **Feasibility** | In a greedy Algorithm, we make whatever choice seems best at the moment in the hope that it will lead to global optimal solution. | In Dynamic Programming we make decision at each step considering current problem and solution to previously solved sub problem to calculate optimal solution . |
| **Optimality** | In Greedy Method, sometimes there is no such guarantee of getting Optimal Solution. | It is guaranteed that Dynamic Programming will generate an optimal solution as it generally considers all possible cases and then choose the best. |
| **Recursion** | A greedy method follows the problem solving heuristic of making the locally optimal choice at each stage. | A Dynamic programming is an algorithmic technique which is usually based on a recurrent formula that uses some previously calculated states. |
| **Memorization** | It is more efficient in terms of memory as it never look back or revise previous choices | It requires dp table for memorization and it increases it’s memory complexity. |
| **Time complexity** | Greedy methods are generally faster. For example, [Dijkstra’s shortest path](https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/) algorithm takes O(ELogV + VLogV) time. | Dynamic Programming is generally slower. For example, [Bellman Ford algorithm](https://www.geeksforgeeks.org/bellman-ford-algorithm-simple-implementation/) takes O(VE) time. |
| **Fashion** | The greedy method computes its solution by making its choices in a serial forward fashion, never looking back or revising previous choices. | Dynamic programming computes its solution bottom up or top down by synthesizing them from smaller optimal sub solutions. |
| **Example** | Fractional knapsack . | 0/1 knapsack problem |