1)What are characteristics of greedy algorithm?

Greedy Algorithms works step-by-step, and always chooses the steps which provide immediate profit/benefit. It chooses the “locally optimal solution”, without thinking about future consequences.

All greedy algorithms follow a basic structure:

Greedy choice property: This property says that the globally optimal solution can be obtained by making a locally optimal solution (Greedy). The choice made by a Greedy algorithm may depend on earlier choices but not on the future. It iteratively makes one Greedy choice after another and reduces the given problem to a smaller one.

Characteristics of Greedy approach

1. There is an ordered list of resources(profit, cost, value, etc.)
2. Maximum of all the resources(max profit, max value, etc.) are taken.
3. For example, in fractional knapsack problem, the maximum value/weight is taken first according to available capacity.

2)Write procedure that describes the general structure of a greedy approach:-

```
getOptimal(Item, arr[], int n)
  1) Initialize empty result : result = {}
  2) While (All items are not considered)

      // We make a greedy choice to select
      // an item.
      i = SelectAnItem()

      // If i is feasible, add i to the
      // result
      if (feasible(i))
        result = result U i
  3) return result
```

Applications of Greedy Algorithms
1. Finding an optimal solution (Activity selection, Fractional Knapsack, Job Sequencing, Huffman Coding).
2. Finding close to the optimal solution for NP-Hard problems like TSP.
Advantages and Disadvantages of Greedy Approach
Advantages

- Greedy approach is easy to implement.
- Typically have less time complexities.
- Greedy algorithms can be used for optimization purposes or finding close to optimization in case of NP Hard problems.

Disadvantages

- The local optimal solution may not always be global optimal.

Why such algorithms are termed as greedy?

In some cases making a decision that looks right at that moment gives the best solution (Greedy), but in other cases it doesn't. The Greedy technique is best suited for looking at the immediate situation

Such algorithms are called greedy because while the optimal solution to each smaller instance will provide an immediate output, the algorithm doesn't consider the larger problem as a whole. Once a decision has been made, it is never reconsidered.

3)Compare and contrast Prim's and Kruskal's Algorithm.

Prim's Algorithm	Kruskal's Algorithm
It starts to build the Minimum Spanning Tree from any vertex in the graph.	It starts to build the Minimum Spanning Tree from the vertex carrying minimum weight in the graph.
It traverses one node more than one	It traverses one node only once.

Prim’s Algorithm	Kruskal’s Algorithm
time to get the minimum distance.	
Prim’s algorithm has a time complexity of $O(V^2)$, V being the number of vertices and can be improved up to O(E log V) using Fibonacci heaps.	Kruskal’s algorithm’s time complexity is O(E log V), V being the number of vertices.
Prim’s algorithm gives connected component as well as it works only on connected graph.	Kruskal’s algorithm can generate forest(disconnected components) at any instant as well as it can work on disconnected components
Prim’s algorithm runs faster in dense graphs.	Kruskal’s algorithm runs faster in sparse graphs.
It generates the minimum spanning tree starting from the root vertex.	It generates the minimum spanning tree starting from the least weighted edge.
Applications of prim’s algorithm are Travelling Salesman Problem, Network for roads and Rail tracks connecting all the cities etc.	Applications of Kruskal algorithm are LAN connection, TV Network etc.

Prims Algorithm	Kruskal Algorithm
It start to build the MST from any of the Node.	It start to build the MST from Minimum weighted vertex in the graph.
Adjencary Matrix , Binary Heap or Fibonacci Heap is used in Prims algorithm	Disjoint Set is used in Kruskal Algorithm.
Prims Algorithm run faster in dense graphs	Kruskal Algorithm run faster in sparse graphs
Time Complexity is O(EV log V) with binay heap and O(E+V log V) with fibonacci heap.	Time Complexity is **O(E log V)**
The next Node included must be connected with the node we traverse	The next edge include may or may hot be connected but should not form the cycle.
It traverse the one node saveral time in order to get it minimum distance	It travese the edge only once and based on cycle it will either reject it or accept it,
Greedy Algorithm	Greedy Algorithm