Greedy Strategy

→ One of the algorithm design strategy
→ For solving optimization problem
→ Algorithm picks the best solution at the moment without regarding consequences
→ It picks the immediate output, but does not consider the big picture, hence it is considered as greedy.
→ It shows an optimal solution that may or may not be true

Applications.

1) Knapsack problem
2) MST
3) Job sequencing
4) Huffman coding
5) Optimal merge pattern
6) Dijkstra's algorithm.

→ Greedy algorithm has 5 components.

1) Candidate set :- A solution is created from this set
2) A selection function :- Used to choose the best candidate to be added to the solution.
3) A feasibility solution :- Used to determine whether a candidate can be used to contribute the solution.
4) An objective function :- Used to assign a value to a solution or a partial solution.
5) A solution function :- Used to indicate whether the complete solution has been reached.

↳ A subset that satisfies the constraints is called a feasible solution.
↳ A feasible solution that maximizes or minimizes a given (objective) function is called Optimal Solution.
↳ The greedy method finds optimal solution from feasible solutions.

Eg:- → Let's take a problem, P.
→ We want to move from location (A) to location (B).
→ But the constraint is that we want to cover this journey in 12 hrs.

P : (A) —12 hrs→ (B)

There are many solutions, we can travel by
→ walk
→ car
→ auto
→ train
→ flight
} but the feasible solutions are by train or by flight.

(Feasible → satisfying the constraint).

→ Let's we want to cover the journey with minimum cost (minimization problem)
→ So out of 2 feasible solutions, the above constraint is satisfied ie, min^m cost or best result
⇒ train → minimum cost satisfying solution.
→ So there can be more than 1 feasible solution, but only one optimal solution

Control Abstraction.
→ Greedy method says that a problem can be solved in stages.
→ In each stage we will consider one input from a given problem and if that i/p is feasible, then we include it in the solution
→ So by including all those input which are feasible we get optimal solution.

Algorithm

```
Algorithm greedy (a, n)
{ for(i=0 to n) do
  { x = Select(a);
    if (feasible(x)) then
      { Solution = Solution + x;
      }
  }
```

a → problem
n → no: of solutions.