Σχεδιάζουμε αλγόριθμο δυναμικού προγραμματισμού σε **προβλήματα** που έχουν τα εξής **χαρακτηριστικά**:

- Ιδιότητα της Άπληστης Επιλογής: Μια ακολουθία άπληστων επιλογών οδηγεί στην βέλτιστη λύση.
- Ιδιότητα των Βέλτιστων Επιμέρους Δομών: Οτί για να λύσουμε το πρόβλημα αρκεί να υπολογίσουμε την βέλτιστη λύση σε κάποια υποπροβλήματα, συνήθως με αναδρομή.

Συνήθης διαδικασία για την κατασκευή ενός άπληστου αλγορίθμου

1. Ταξινομούμε τα δεδομένα από τα οποία επιλέγουμε την λύση
2. Επιλέγουμε το επόμενο στοιχείο με βάση την ταξινόμηση για να το εισάγουμε στη λύση μας.
    1. Αν η λύση που προκύπτει δεν παραβιάζει τους περιορισμούς του προβλήματος, διατηρούμε το στοιχείο στη λύση
    2. Αν η λύση που προκύπτει παραβιάζει τους περιορισμούς του προβλήματος, τότε απορρίπτουμε το στοιχείο.

   Εωσότου κατασκευαστεί η λύση

Ένας Άπληστος Αλγόριθμος:

Μπορεί να είναι βέλτιστος: Η απόδειξη γίνεται με δύο εναλλακτικούς (και συμπληρωματικούς) τρόπους:

1. Με μαθηματική επαγωγή. Ότι κάθε επιλογή του άπληστου αλγορίθμου είναι βέλτιστη.
2. Με απόδειξη των δύο ιδιοτήτων (βέλτιστες επιμέρους δομές και άπληστη επιλογή)

Μπορεί να μην είναι βέλτιστος: Η απόδειξη γίνεται με κατάλληλο αντιπαράδειγμα:

1. Δείχνουμε ότι ο αλγόριθμος επιστρέφει μία λύση που έχει ένα κόστος, που είναι χειρότερο από
2. Την βέλτιστη λύση.

Παραδείγματα Άπληστων Αλγορίθμων:

1. **Αγλόριθμος Dijkstra** για υπολογισμό συντομότερων μονοπατιών σε γράφημα: Πολυπλοκότητα: $O(n^2)$ και με ειδική δομή δεδομένων: O(m+nlogn)
2. **Αλγόριθμος Prim** για υπολογισμό συνδετικού δένδρου ελαχίστου κόστους: Πολυπλοκότητα: $O(n^2)$ και με ειδική δομή δεδομένων: O(m+nlogn)
3. **Αλγόριθμος Kruskal** για υπολογισμό συνδετικού δένδρου ελαχίστου κόστους: Πολυπλοκότητα O(m logn)
4. **Επιστροφή Ρέστων.** Πολυπλοκότητα: O(X), όπου X το ποσό επιστροφής