FICHE DE REVISION - TERMINALE NSI

---

1. Récursivité

Definition : Une fonction qui s'appelle elle-même pour résoudre un problème.
Utilité : Décomposer un problème complexe en plusieurs versions plus simples de lui-même.
Avantages : Solutions élégantes, adaptées aux structures imbriquées (arbres, fractales).
Inconvénients : Risque de boucle infinie, consomme plus de mémoire que l'itératif.
Exemple concret : Pour calculer une opération sur une liste, on traite le premier élément, et on demande à soi-même de traiter le reste de la liste.

2. Diviser pour régner

Principe : Diviser un problème en sous-problèmes plus petits, résoudre chacun, puis combiner.
Utilité : Accélérer le traitement de problèmes volumineux.
Avantages : Très efficace, souvent utilisé dans des algorithmes rapides.
Inconvénients : Plus difficile à comprendre, nécessite souvent de la récursivité.
Exemple concret : On découpe une liste en 2 moitiés, on traite chaque moitié, puis on réassemble.

3. Tri fusion (Merge Sort)

Principe : 1) Diviser la liste en deux, 2) Trier chaque moitié, 3) Fusionner proprement les deux listes triées.
Utilité : Trier rapidement des grandes listes.
Avantages : Très rapide et constant, complexité O(n log n).
Inconvénients : Utilise plus de mémoire car il faut stocker des sous-listes.
Exemple concret : Comme découper une pile de papiers en petites piles, les trier, et les recombiner ordonné

4. Algorithmes de première

4.1. Dichotomie

Principe : Chercher dans une liste triée en divisant l'intervalle en deux à chaque étape.
Avantages : Très rapide, O(log n).
Inconvénients : Liste obligatoirement triée.
Exemple : Chercher un mot dans un dictionnaire en ouvrant au milieu.

4.2. Tri par sélection

Principe : Rechercher le plus petit élément et le placer au début, répété n fois.
Avantages : Simple à comprendre.
Inconvénients : Très lent, O(n^2), beaucoup de comparaisons.
Exemple : Trier des cartes en cherchant toujours la plus petite.

4.3. Tri par insertion

Principe : Construire progressivement une liste triée en insérant chaque élément à la bonne place.
Avantages : Très efficace si la liste est presque triée.
Inconvénients : Lent dans le pire cas, O(n^2).
Exemple : Ranger des feuilles dans un classeur déjà trie.

5. Complexités (comparaisons)
Tri fusion : $O(n \log n) \rightarrow$ très rapide.
Dichotomie : $O(\log n) \rightarrow$ extrêmement rapide.
Tri sélection : $O(n^2) \rightarrow$ lent.
Tri insertion : $O(n^2)$ pire cas, $O(n)$ meilleur cas.

6. Exemples d'utilisation concrète
- Récursivité : traitement d'arbres, fractales, exploration de dossiers.
- Diviser pour régner : calcul efficace sur de grandes données.
- Tri fusion : trier rapidement de gros volumes.
- Dichotomie : recherche ultra-rapide dans une base triée.