Chapter II : The Lists

Introduction

Recursion is a fundamental concept in computer science where a function calls itself to break down a problem into smaller subproblems. This approach is particularly useful in problems that exhibit **self-similarity** and can be broken down into simpler versions of the same problem.

Principle of Lists in Algorithms and Data Structures

A list is a fundamental data structure that represents an ordered collection of elements. Lists can store multiple values of the same or different types and allow various operations such as insertion, deletion, traversal, and searching.

1. Basic Concept of Lists

A **list** is a fundamental data structure that represents an ordered collection of elements. Lists can store multiple values of the same or different types and allow various operations such as insertion, deletion, traversal, and searching.

2. Types of Lists

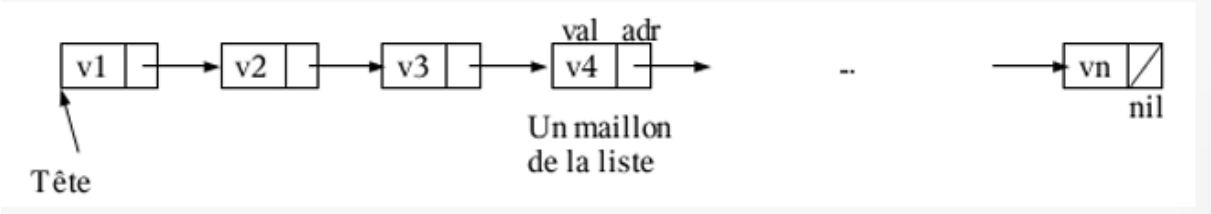
A. Linear Lists

Array (Static List)

- A contiguous block of memory storing elements.
- Fast random access (O(1)).
- Fixed size (in static arrays).

Linked List (Dynamic List)

- Consists of nodes with pointers linking them.
- Supports dynamic resizing.
- Variants:
  - **Singly Linked List** (each node points to the next)
  - **Doubly Linked List** (each node points to the next and previous)
  - **Circular Linked List** (last node connects back to the first)



Definition of a List in C

In **C programming**, a **list** is a data structure used to store multiple elements dynamically. Since C does not have a built-in list type like Python or Java, lists are typically implemented using **arrays** or **linked lists**.

Implementing a List in C

A. Using Arrays (Static List)

An **array-based list** is a simple way to store elements in contiguous memory locations.

Example: List using Arrays

```c
#include <stdio.h>

#define SIZE 10  // Define maximum size of the list

int main() {
    int list[SIZE] = {10, 20, 30, 40, 50}; // Initialize list with values
    int length = 5; // Current number of elements

    // Display elements
    printf("List elements: ");
    for (int i = 0; i < length; i++) {
        printf("%d ", list[i]);
    }
    printf("\n");

    return 0;
}
```