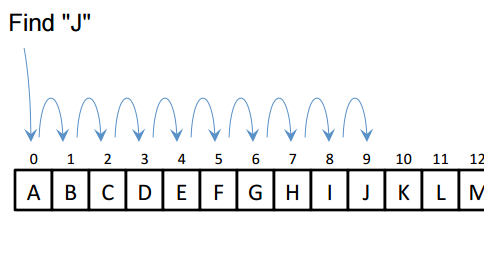
# MỘT SỐ THUẬT TOÁN TÌM KIẾM

# I.Thuật toán tìm kiếm tuyến tính(**Linear Search):**

## 1.Tìm kiếm tuyến tính là gì?

Tìm kiếm tuyến tính là phương pháp tìm kiếm một phần tử cho trước trong một danh sách bằng cách duyệt lần lượt từng phần từ của danh sách đó đến khi nào tìm được giá trị mong muốn hay đã duyệt hết qua hết danh sách.



## 2.Ý tưởng thuật toán:

Sau đây là ý tưởng triển khai thuật toán:

* Bắt đầu từ bản ghi đầu tiên của mảng, duyệt từ đầu mảng đến cuối mảng với x.
* Nếu phần tử đang duyệt bằng x thì trả về vị trí.
* Nếu không tìm thấy bất cứ phần từ nào khi đã duyệt hết thì trả về -1.

## 3.Độ phức tạp thuật toán:

Độ phức tạp thuật toán là O(n), với n là số lượng phần tử trong dãy A[].

## 4.Ưu ,nhược điểm:

### 4.1.Ưu điểm:

* **Đơn giản và dễ hiểu**: Thuật toán dễ triển khai và không yêu cầu mảng phải được sắp xếp.
* **Không cần cấu trúc dữ liệu đặc biệt**: Có thể áp dụng trên bất kỳ danh sách hoặc mảng nào.

### 4.2.Nhược điểm:

* **Hiệu suất kém với mảng lớn**: Do phải duyệt qua từng phần tử, thời gian tìm kiếm tăng lên theo kích thước của mảng.
* **Không tối ưu**: Không hiệu quả cho việc tìm kiếm trong mảng hoặc danh sách lớn.

## 5.Cài đặt thuật toán:

A black background with white text

Description automatically generated

# II.Thuật toán tìm kiếm nhị phân(Binary Search):

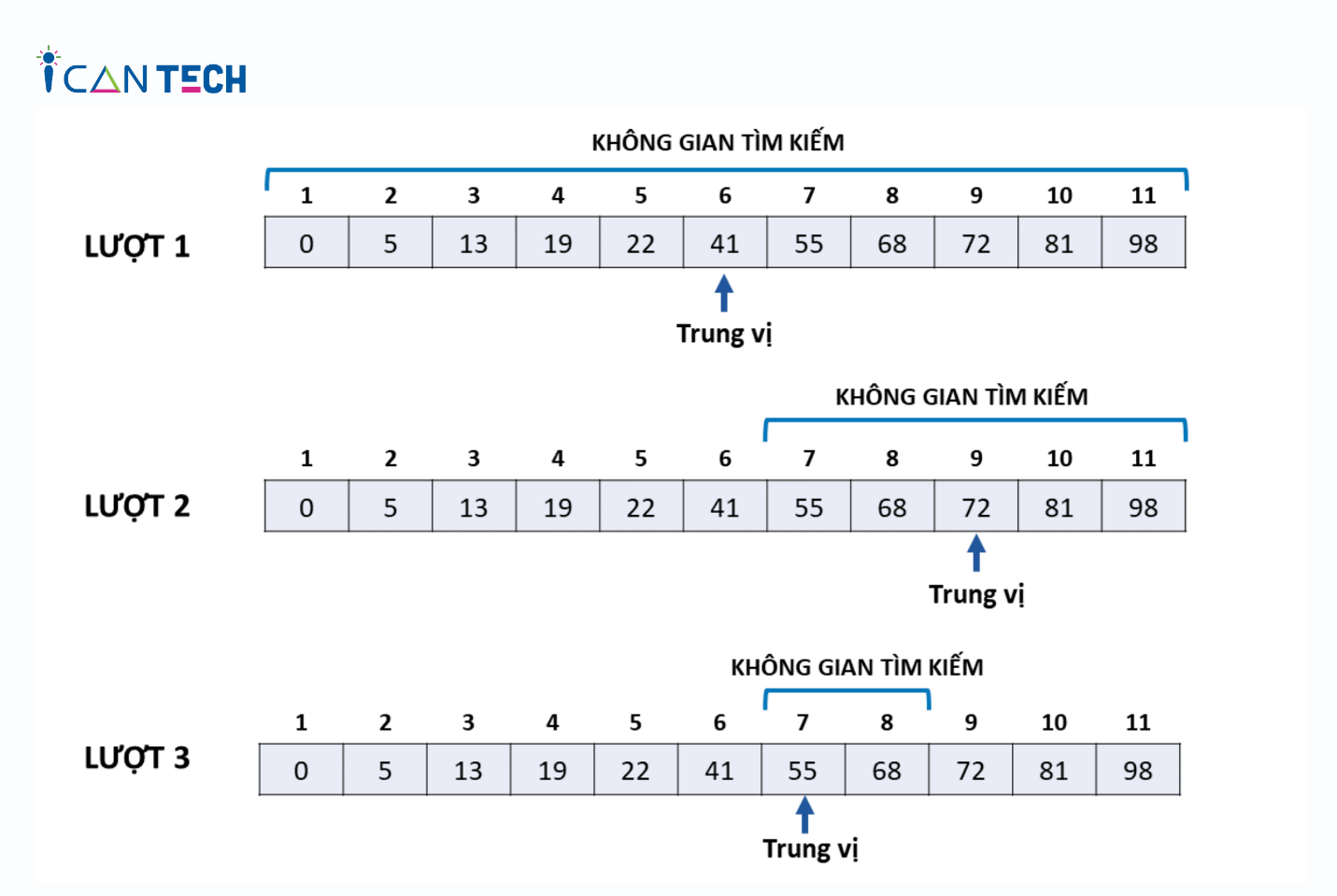
## 1.Tìm kiếm nhị phân là gì ?

Thuật toán tìm kiếm nhị phân là phương pháp định vị phần tử *x* trong một danh sách A[] gồm n phần tử đã được sắp xếp.

## 2.Ý tưởng thuật toán tìm kiếm nhị phân:

Ý tưởng tìm kiếm nhị phân là chia tập dữ liệu thành các phần nhỏ hơn và loại trừ một nửa của chúng sau mỗi bước.

Dưới đây là các bước cơ bản của thuật toán tìm kiếm nhị phân:



* Bước 1: Xác định phạm vi tìm kiếm: Bắt đầu bằng việc xác định phạm vi tìm kiếm ban đầu. Thông thường, phạm vi này ban đầu sẽ là toàn bộ tập dữ liệu (từ vị trí đầu tiên đến vị trí cuối).
* Bước 2: Tìm phần tử giữa: Tính toán chỉ số của phần tử ở giữa trong phạm vi tìm kiếm. Điều này thường được thực hiện bằng cách lấy trung bình của chỉ số đầu và chỉ số cuối.
* Bước 3: So sánh với phần tử giữa: So sánh phần tử tại vị trí giữa với giá trị cần tìm kiếm.

- Nếu phần tử giữa bằng giá trị cần tìm, thì tìm thấy và kết thúc tìm kiếm.

- Nếu phần tử giữa lớn hơn giá trị cần tìm, điều này có nghĩa rằng phần tử cần tìm nằm ở phía bên trái của phần tử giữa, vì tập dữ liệu đã sắp xếp. Do đó, thuật toán sẽ cập nhật phạm vi tìm kiếm cho phần bên trái của phần tử giữa và lặp lại quy trình trên phạm vi này.

- Nếu phần tử giữa nhỏ hơn giá trị cần tìm, tương tự, thuật toán sẽ cập nhật phạm vi tìm kiếm cho phần bên phải của phần tử giữa và tiếp tục tìm kiếm trên phạm vi này.

* Bước 4: Lặp lại quy trình: Lặp lại bước 2 và bước 3 cho đến khi tìm thấy phần tử cần tìm hoặc phạm vi tìm kiếm thu nhỏ đến một mức không thể thu nhỏ nữa.
* Bước 5: Kết thúc: Khi tìm thấy phần tử cần tìm hoặc phạm vi tìm kiếm đã thu nhỏ đến mức tối thiểu, thuật toán kết thúc.

## 3.Ưu ,nhược điểm:

### 3.1.Ưu điểm:

* **Hiệu suất cao**: Tìm kiếm nhanh chóng trong mảng đã được sắp xếp, giảm đáng kể số lần so sánh.
* **Hiệu quả**: Độ phức tạp thấp hơn nhiều so với Linear Search.

### 3.2Nhược điểm:

* **Yêu cầu mảng đã sắp xếp**: Phải sắp xếp mảng trước khi tìm kiếm.
* **Phức tạp hơn**: Cài đặt phức tạp hơn so với Linear Search.

## 4.Độ phức tạp:

* Độ phức tạp thuật toán là O(log(n)), với n là số lượng phần tử của dãy A[].

## 5.Cài đặt thuật toán:

A computer screen with white text

Description automatically generated

# III.So sánh thuật toán tìm kiếm nhị phân và thuật toán tìm kiếm tuyến tính:

|  |  |  |
| --- | --- | --- |
| Đặc điểm | Tìm kiếm tuyến tính | Tìm kiếm nhị phân |
| Ý tưởng | Duyệt qua từng phần tử tuần tự từ đầu đến cuối cho đến khi tìm thấy phần tử cần tìm hoặc hết mảng. | Chia đôi phạm vi tìm kiếm, so sánh phần tử ở giữa với phần tử cần tìm, và tìm kiếm trong nửa phù hợp. |
| Yêu cầu mảng sắp xếp | Không | Có |
| Ưu điểm | Đơn giản, dễ hiểu, không yêu cầu mảng sắp xếp, có thể áp dụng trên bất kỳ danh sách nào. | Hiệu suất cao, tìm kiếm nhanh trong mảng đã sắp xếp, số lần so sánh ít. |
| Nhược điểm | Hiệu suất kém với mảng lớn, thời gian tìm kiếm tăng theo kích thước mảng. | Yêu cầu mảng đã sắp xếp trước, cài đặt phức tạp hơn. |
| Ứng dụng phù hợp | Khi mảng nhỏ hoặc không sắp xếp, hoặc khi tìm kiếm chỉ thực hiện một lần. | Khi cần tìm kiếm nhiều lần trên mảng đã sắp xếp, hoặc khi mảng lớn. |