**Câu 1: Trình bày ngắn gọn tư tưởng các giải thuật tìm kiếm, các giải thuật này có thể được vận dụng trong các trường hợp nào, cho ví dụ minh họa.**

- Linear Search (Tìm kiếm tuyến tính): là một giải thuật đơn giản, rất dễ cài đặt. Bắt đầu từ đối tượng a1, duyệt qua tất cả các đối tượng, cho tới khi tìm thấy đối tượng có khóa mong muốn, hoặc duyệt hết toàn bộ dãy mà không tìm thấy khóa đó.

+ Mặc dù giải thuật Tìm kiếm tuyến tính rất đơn giản và dễ cài đặt, tuy nhiên nhược điểm của nó nằm ở độ phức tạp. Trong trường hợp tốt nhất, giải thuật có độ phức tạp là O(1), nhưng trong trường hợp xấu nhất lên tới O(n). Vì vậy độ phức tạp tổng quát của giải thuật là O(n), chỉ phù hợp với những bài toán có kích thước không gian tìm kiếm nhỏ.

+ Ví dụ: Cho một dãy số a gồm n số nguyên a1, a2, … an ( 1 ≤ n ≤ 1000 ). Hãy xác định xem số fibonacci thứ k (1 ≤ k ≤ 100) có xuất iện trong dãy số hay không, nếu có thì đưa ra vị trí xuất hiện đầu tiên, ngược lại đưa ra −1.

- Binary Search (Tìm kiếm nhị phân): Không gian tìm kiếm cần được sắp xếp lại theo chiều tăng dần hoặc giảm dần của khóa tìm kiếm (mục tiêu là để tạo ra dãy có tính thứ tự). Chọn am ở giữa A để tận dụng kết quả so sánh với khóa x. A được chia thành hai phần: trước và sau am. Chỉ số bắt đầu, kết thúc của A là l và r. Nếu x = am, tìm thấy và dừng. Xét thứ tự x, am. Nếu thứ tự này là R, thì tìm x trong đoạn [l, r] với r=m-1; Ngược lại, tìm x trong đoạn [l, r] với l=m+1.

+ Giải thuật Tìm kiếm nhị phân chỉ có thể thực hiện trên một tập đã sắp xếp, chính vì thế chi phí sắp xếp cũng cần được tính đến. Nếu như dãy số bị thay đổi bởi các thao tác thêm, xóa hay sửa phần tử, thì việc sắp xếp cũng phải thực hiện lại liên tục, từ đó dẫn đến thời gian thực thi bị tăng lên.

+ Ví dụ: ho dãy số A gồm n (1≤ n ≤105) phần tử nguyên a1, a2 ... an ​ (ai ≤ 109). hãy xác định số chính phương nhỏ nhất không xuất hiện trong dãy số?