BÀI 7

LẬP TRÌNH GIẢI BÀI TOÁN TÌM KIẾM

Học xong bài này, em sẽ:

- Phát biểu được bài toán tìm kiếm.
- Viết được chương trình cho một số thuật toán tìm kiếm.
- Vận dụng được quy tắc thực hành xác định được độ phức tạp của một vài thuật toán tìm kiếm đơn giản.

Khi tạo mới một tài khoản người dùng, em được yêu cầu nhập tên người dùng "user name". Có trường hợp em phải nhập lại tên khác vì tên vừa nhập đã có người sử dụng rồi. Theo em, máy tính làm gì ngay sau khi nhận được yêu cầu tạo mới một tài khoản? Hãy phát biểu thành một bài toán.

1 Bài toán tìm kiếm

a) Khái niệm bài toán tìm kiếm

Các ví dụ thực tế dẫn đến bài toán tìm kiếm:

– Cho mã cuốn sách, hãy tìm cuốn sách trong kho sách của thư viện.

– Tìm một tên người, tên hàng hoá,... trong danh sách liệt kê.

– Tìm bản ghi có khoá là k trong bảng T của một cơ sở dữ liệu.

Theo nghĩa chung nhất, bài toán tìm kiếm là: Cho một *yêu cầu tìm kiếm* và một tập hợp dữ liệu là *phạm vi tìm kiếm*. Hãy tìm mục (các mục) dữ liệu đáp ứng yêu cầu tìm kiếm đã cho hoặc khẳng định không có mục dữ liệu nào đáp ứng yêu cầu đó.

Tuỳ theo yêu cầu và phạm vi tìm kiếm mà bài toán tìm kiếm là dễ hay khó. Một nhiệm vụ của máy tìm kiếm trên Internet nhằm giải quyết bài toán tìm kiếm có phạm vi tìm kiếm là dữ liệu văn bản (hình ảnh hoặc giọng nói) trên Internet.

b) Tìm kiếm tuần tự bằng hàm của Python

Python có phương thức `index` thực hiện tìm kiếm phần tử x trong một dãy tuần tự (xâu kí tự, mảng hoặc danh sách) và trả về:

– Nếu xuất hiện nhiều lần thì đưa ra chỉ số của lần xuất hiện đầu tiên.

Đọc bản mới nhất trên hoc10.vn Bản sách mẫu

– Báo lỗi "**ValueError**" nếu không tìm thấy.

– Phương thức **index** có hai tham số tuỳ chọn: **lo**, **hi** để hạn chế thực hiện tìm kiếm chỉ trong đoạn con của dãy số, bắt đầu từ chỉ số **lo** (**lowest**) và kết thúc ở **hi** (**highest**). Cú pháp:

```
dãy_số.index(giá_trị, lo, hi)
```

Ví dụ: Với mảng a = [1, 2, 3, 4, 5, 6] như hình bên, câu lệnh **print(a.index(3,1,4))** sẽ in ra màn hình kết quả là 2, cho biết vị trí của phần tử 3 trong đoạn [1, 4] ở mảng a.

```
File  Edit  Format  Run  Options  Window  Help
a = [1, 2, 3, 4, 5, 6]
print(a.index(3, 1, 4))
```

2 Thuật toán tìm kiếm tuần tự

Chi tiết dần từng bước thuật toán tìm kiếm tuần tự

Hình 1 mô tả liệt kê các bước của thuật toán tìm kiếm tuần tự một số x:

Số đang xét là số ở đầu dãy
Lặp khi *Chưa hết dãy số*:
    Nếu *Số đang xét* ≠ x:
        Chuyển đến xét số tiếp theo trong dãy
    Trái lại:
        Thông báo vị trí tìm thấy x và kết thúc
    Hết nhánh
Hết lặp
Thông báo không tìm thấy x và kết thúc

Hình 1. Thuật toán tìm kiếm tuần tự

Trong mô tả trên có một số cụm từ như: "*Số đang xét* là số ở đầu dãy", "*Chưa hết dãy số*", "Chuyển đến xét số tiếp theo trong dãy", "Thông báo vị trí tìm thấy x và kết thúc", "Thông báo không tìm thấy x và kết thúc". *Hình 2* là kết quả thay thế những cụm từ trên bằng mã giả.

```
i ← 0                          # Số đang xét là a_0 ở đầu dãy
while (i < n):                 # (i < n) tức là chưa hết dãy số
    if a_i ≠ x :
        i ← i + 1              # Chuyển đến xét số tiếp theo
    else:
        return i               # Đã tìm thấy
return không tìm thấy
```

Hình 2. Mã giả của thuật toán tìm kiếm tuần tự

Đọc bản mới nhất trên hoc10.vn Bản sách mẫu

3 Thuật toán tìm kiếm nhị phân

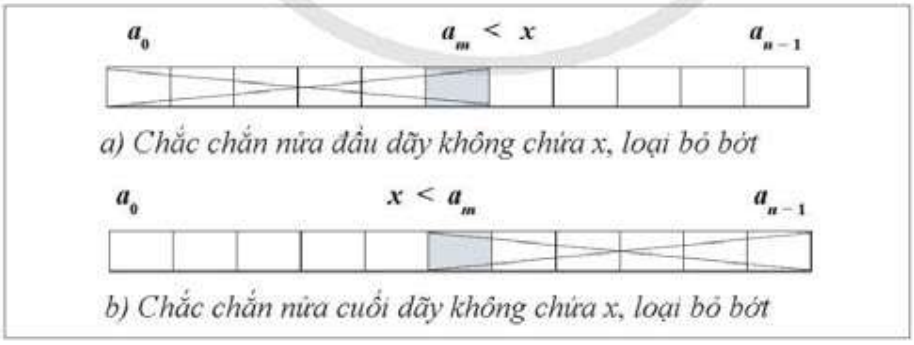
Dựa trên mô tả thuật toán tìm kiếm nhị phân cho ở *Hình 3*, em hãy nêu tóm tắt ý tưởng của thuật toán này.

Nếu dãy số đã sắp thứ tự thì có thể áp dụng thuật toán tìm kiếm nhị phân như mô tả trong *Hình 3*:

Xuất phát: *Phạm vi tìm kiếm là dãy ban đầu*
Lặp khi *Vẫn còn Phạm vi tìm kiếm* **:**
Xác định phần tử a_m ở giữa Phạm vi tìm kiếm
Nếu $x = a_m$ **:**
Thông báo tìm thấy x ở vị trí m và kết thúc
Trái lại **:**
Loại bỏ nửa dãy chắc chắn không chứa x
Phạm vi tìm kiếm là nửa dãy còn lại
Hết nhánh
Hết lặp
Thông báo không tìm thấy x và kết thúc

Hình 3. Liệt kê các bước của thuật toán tìm kiếm nhị phân

Hình 4 minh hoạ cho dãy sắp thứ tự tăng dần (không giảm):



Hình 4. Sơ đồ một bước thuật toán tìm kiếm nhị phân

Hướng dẫn viết mã giả của thuật toán tìm kiếm nhị phân

– Xác định các *cụm từ* cần làm chi tiết hơn bằng mã giả: "*Phạm vi tìm kiếm là dãy ban đầu*"; "*Vẫn còn phạm vi tìm kiếm*"; "*Xác định phần tử a_m ở giữa phạm vi tìm kiếm*",

Đọc bản mới nhất trên hoc10.vn Bản sách mẫu

"*Loại bỏ nửa dãy chắc chắn không chứa x*", "*Phạm vi tìm kiếm là nửa dãy còn lại*", "*Thông báo không tìm thấy x và kết thúc*".

– Bổ sung thêm `lo` là chỉ số phần tử ở đầu trái đoạn con và `hi` là chỉ số phần tử ở đầu phải đoạn con.

– Công thức tính chỉ số *m* của phần tử ở "giữa" đoạn con là (`lo` + `hi`)//2, kết quả đảm bảo là số nguyên.

4 Thực hành lập trình giải bài toán tìm kiếm

Nhiệm vụ 1. Em hãy thực hiện các yêu cầu sau:

a) Viết mã giả cho thuật toán tìm kiếm nhị phân.

b) Ước lượng số lần thực hiện vòng lặp trong thuật toán tìm kiếm nhị phân.

c) Ước lượng độ phức tạp thời gian của thuật toán tìm kiếm nhị phân.

Hướng dẫn: Sau lần chia đôi đầu tiên; phạm vi tìm kiếm còn lại $\frac{n}{2}$ số; sau khi chia đôi lần thứ hai, dãy còn lại $\frac{n}{4}$ số; sau khi chia đôi lần thứ ba, dãy còn lại $\frac{n}{8}$ số,... sau khi chia đôi lần *k* dãy còn lại $\frac{n}{2^k}$ số. Kết thúc khi $2^k \sim n$.

Nhiệm vụ 2. Em hãy thực hiện các yêu cầu sau:

a) Viết chương trình Python thực hiện tìm kiếm tuần tự.

b) Viết phiên bản tìm kiếm tuần tự thứ hai, dùng vòng lặp `for` thay cho vòng lặp `while` (hoặc ngược lại).

c) Viết phiên bản tìm kiếm tuần tự có thêm hai tham số đầu vào `lo` và `hi` tương tự như của hàm `index`. So sánh kết quả với phương thức `index` của Python.

Nhiệm vụ 3. Viết hàm thực hiện tìm kiếm nhị phân nhận hai tham số đầu vào: dãy số *a* và giá trị *x* cần tìm.

Viết chương trình tìm kiếm vị trí tên của một người trong mỗi danh sách sau đây:

a) Danh sách học sinh của lớp em.

b) Danh sách tên các chủ tài khoản ngân hàng (kí tự không dấu) và đã sắp thứ tự theo bảng chữ cái.

Câu 1. Em hãy nêu ra một vài ví dụ về bài toán tìm kiếm trong thực tế.

Câu 2. Theo em, với dãy đã sắp thứ tự và cho một số *x* cụ thể:

a) Trường hợp nào tìm kiếm tuần tự nhanh hơn tìm kiếm nhị phân?

b) Về trung bình thuật toán tìm kiếm tuần tự hay thuật toán tìm kiếm nhị phân tốt hơn?

Đọc bản mới nhất trên hoc10.vn Bản sách mẫu

Tóm tắt bài học

- Thực hiện tìm kiếm tuần tự bằng phép lặp duyệt từ đầu dãy số với điều kiện dừng khi "tìm thấy" hoặc "đã xét hết dãy số" .
- Phép lặp thực hiện tìm kiếm nhị phân chia đôi dãy số tại điểm "giữa" có chỉ số (`lo + hi`)//2, bỏ bớt nửa dãy cho đến khi "tìm thấy" hoặc hết dãy.

BÀI TÌM HIỂU THÊM

CÁC THAM SỐ TUỲ CHỌN CHO MỘT HÀM

Các hàm thực hiện tìm kiếm của Python đều có các tham số tuỳ chọn `lo` và `hi` để xác định một phạm vi tìm kiếm là đoạn con của dãy số đầu vào. Ta đã viết các hàm thực hiện tìm kiếm có thêm hai tham số `lo` và `hi` có vai trò tương tự. Tuy nhiên, đây không phải là những tham số tuỳ chọn vì chúng bắt buộc phải có trong lời gọi sử dụng các hàm này. Làm thế nào để chúng trở thành *tuỳ chọn*?

Một giải pháp là dùng *bộ* (*tuple*) *các tham số* `*args` trong định nghĩa hàm.

1) Dấu sao (`*`) là kí hiệu toán tử, sau đó là một tên bộ, gộp nhiều tham số. Có thể tuỳ ý đặt tên bộ; `args` chỉ là thông lệ thường thấy.

2) Quy định `*args` phải đứng sau tất cả các tham số bắt buộc của hàm.

Theo mẫu trong *Hình 5*, ta có thể định nghĩa các hàm có tham số tuỳ chọn.

File Edit Format Run Options Window Help

```
def tkNhiPhan(a,x,*args):
    #Nếu có đối số trong lời gọi hàm
    if len(args) > 0:
        lo = args[0]
        hi = args[1]
    #Nếu không có trong lời gọi hàm
    else:
        lo = 0 #Giá trị mặc định
        hi = len(a)- 1
    while (lo <= hi):...
```

Hình 5. Một hàm tự định nghĩa có tham số tuỳ chọn

Đọc bản mới nhất trên hoc10.vn Bản sách mẫu