Chuyên đề:Interval Tree

0. Bài toán tiếp cận:

Có nhân viên, đến ngày phát lương thì lương của nhân viên thứ là . Nhưng nhân viên nhận nhiệm vụ phát lương là người mới, nên chưa có nhiều kinh nghiệm và khá hậu đậu. Anh ta không biết mức lương mình đã phát là bao nhiêu và đôi khi còn phát thừa hoặc thiếu lương nữa. Hãy giúp anh ta biết tổng số lương đã phát cho nhân viên từ đến là bao nhiêu và cập nhật lại nếu cần.

INP: Dòng đầu chứa 2 số và . Dòng sau chứa số lần lượt là . dòng tiếp theo, mỗi dòng chứa 3 số . Nếu thì in ra tổng lương của các nhân viên từ đến . Nếu thì cập nhật lại lương của nhân viên là (tức ).

OUT: Nhiều dòng là kết quả của các truy vấn .

Sample:

INP:

5 3

1 2 3 4 5

1 1 4

2 5 6

1 2 5

OUT:

10

15

- Subtask 1: .

- Subtask 2: . Không có truy vấn loại 2.

- Subtask 3: .

***Giải:***

Một cách làm dễ nhìn thấy đó chính là với mỗi truy vấn loại 1, chúng ta chỉ cần duyệt qua các phần tử từ đến , tính tổng và xuất ra kết quả.

Với cách duyệt này, chúng ta chỉ có thể làm được *Subtask 1*.

Cách làm trên có một hạn chế, đó là sẽ có lúc chúng ta tính lại những đoạn mà chúng ta đã xét rồi. Vậy để rút gọn thời gian tính toán, chúng ta có thể ghi nhớ lại những giá trị bằng cách tính một mảng , với là tổng các số từ 1 đến . Như vậy với mỗi truy vấn, chúng ta chỉ cần tốn để tính được tổng bằng công thức:

.

Mặt khác, với mỗi truy vấn loại 2, với yêu cầu cập nhật lại giá trị phần tử trong mảng, ta lại phải cập nhật lại mảng , vì vậy chúng ta chỉ có thể làm được *Subtask 2.*

Vậy chúng ta phải tiếp cận bằng một hướng khác để giải quyết được *Subtask 3*.

Hạn chế của cách 1 là phải duyệt lại những thành phần đã duyệt qua 1 lần rồi, cách giải quyết vấn đề này chính là sử dụng những biến nhớ lại kết quả đã tính. Nhưng hạn chế của việc xài biến nhớ chính là sau mỗi lần cập nhật giá trị 1 biến lại phải tính lại hầu như tất cả biến nhớ.

Để ý rằng, nếu đã có được mảng nhớ của phần tử rồi, cập nhật lại giá trị bất kỳ, giả sử cần tính đoạn từ đến , ta hoàn toàn có thể tính đoạn từ đến rồi cộng thêm phần tử mới để có kết quả đúng. Tức là, chúng ta có thể chỉ cập nhật lại một vài khoảng nhất định, rồi lấy khoảng mới cập nhật đó hợp với những khoảng đã xét, để ra kết quả đúng.

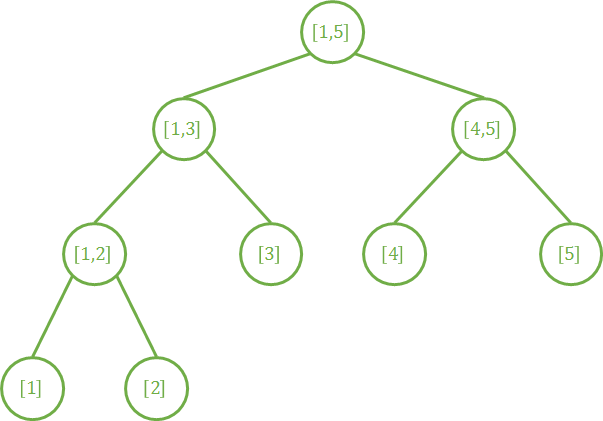
Vấn đề là, cập nhật như thế nào là tiện nhất, độ phức tạp ổn định nhất và dễ cài đặt nhất. Về vấn đề này, chúng ta sẽ cùng nhau điểm qua một cấu trúc dữ liệu mới: **Interval Tree.**

**1. Giới thiệu:**

Cây Interval Tree – IT (hay còn gọi là cây Segment Tree - ST) là một cấu trúc dữ liệu được sử dụng rất nhiều, đặc biệt là trong các bài toán xử lý trên dãy số.

IT là một cây nhị phân đầy đủ (mỗi nút là lá hoặc có 2 nút con) với mỗi nút của cây sẽ quản lý một dãy các phần tử liên tiếp.

Với một dãy số gồm phần tử, nút gốc sẽ lưu thông tin về đoạn , nút con trái của nó sẽ lưu thông tin về đoạn và nút con phải sẽ lưu thông tin về đoạn . Nói một cách tổng quát: nếu nút lưu thông tin đoạn thì 2 nút con và sẽ lưu thông tin của các đoạn và đoạn .



2. Cài đặt:

Chúng ta có thể cài đặt bằng mảng 1 chiều, phần tử thứ nhất của mảng thể hiện nút gốc. Phần tử thứ sẽ có 2 phần tử con là (nút con trái) và (nút con phải), và có phần tử cha là .



Với cách cài đặt này, người ta đã chứng minh được bộ nhớ cần dùng cho IT không vượt quá phần tử.

*- Dựng cây:* Gọi thủ tục → Dựng cây IT gốc quản lý đoạn .

1. Lưu trữ phạm vi: .

2. Nếu → là nút lá:

→ Cập nhật (tức nút lá chứa duy nhất một giá trị phần tử là nút ).

→ Lưu trữ thông tin của phần tử vào nút .

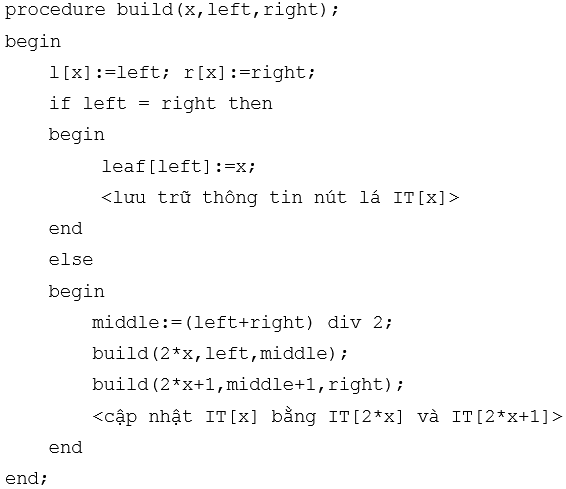
3. Nếu → là nhánh:

→ Tính .

→ Dựng cây IT từ nút con trái → Gọi .

→ Dựng cây IT từ nút con phải → Gọi .

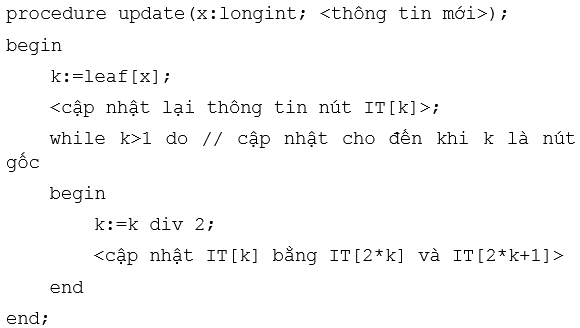
→ Cập nhật thông tin của đoạn bằng 2 nhánh con và .



*- Cập nhật:* Thay đổi phần tử thứ trong dãy.

→ Cập nhật lại thông tin nút lá .

→ Cập nhật lại thông tin các nút cha của .



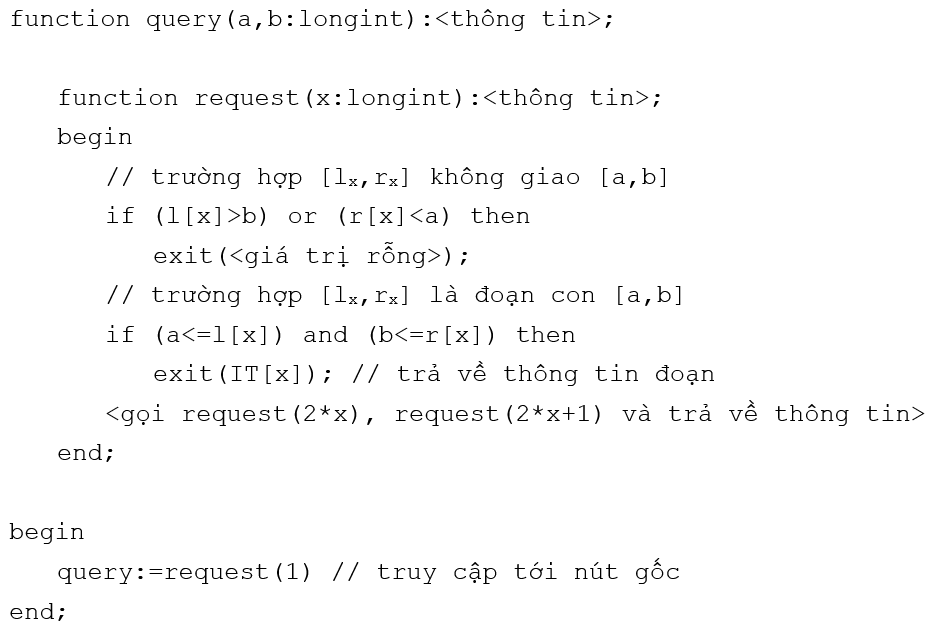
*- Trả lời truy vấn:* Trả lời thông tin đoạn → Truy cập tới từng đoạn con để xét:

→ : Truy cập tới nút gốc chứa đoạn để xét.

→ Nếu , trả về giá trị rỗng (không làm ảnh hưởng đến việc xét các đoạn khác).

→ Nếu , tức là đoạn mà nút quản lý là một phần của đoạn đang xét , nên ta trả về thông tin của đoạn , tức .

→ Trong trường hợp còn lại, tức , ta truy cập tiếp đến 2 nút con và để lấy thông tin và loại trừ đoạn không thuộc .



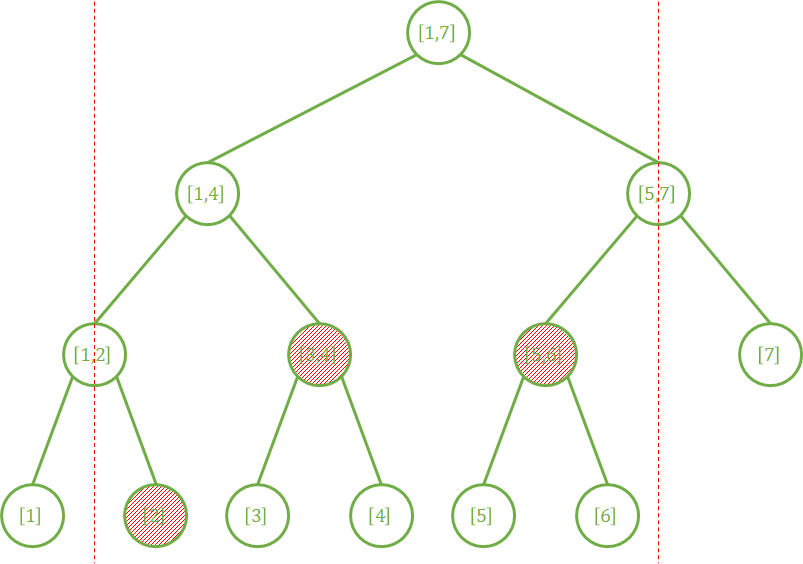
3. Phân tích thời gian chạy:

- Ở thao tác dựng cây, chiều rộng của cây là và chiều cao là , vì vậy tốn .

- Ở thao tác cập nhật, tìm được nút lá chứa thông tin trực tiếp của phần tử trong và cập nhật các nút cha tốn khoảng (chiều cao của cây là ), vì vậy thao tác này tốn khoảng ).

- Ở thao tác trả lời truy vấn, có thể thấy thao tác chỉ tốn khoảng .

Ví dụ trong một dãy có 7 phần tử, xét lấy thông tin đoạn :



Có thể thấy để lấy thông tin đoạn , đối với cây IT chỉ cần truy cập tới 3 nút là được, vì đoạn và đã được tích hợp thông tin từ lúc dựng cây (và lúc cập nhật nếu có) nên không cần xét đến các nút con. Tổng hợp thông tin của 3 đoạn này là thông tin của đoạn .

Vậy, thao tác trả lời truy vấn tốn khoảng .

4. Phân tích bộ nhớ:

Ta xét 2 trường hợp:

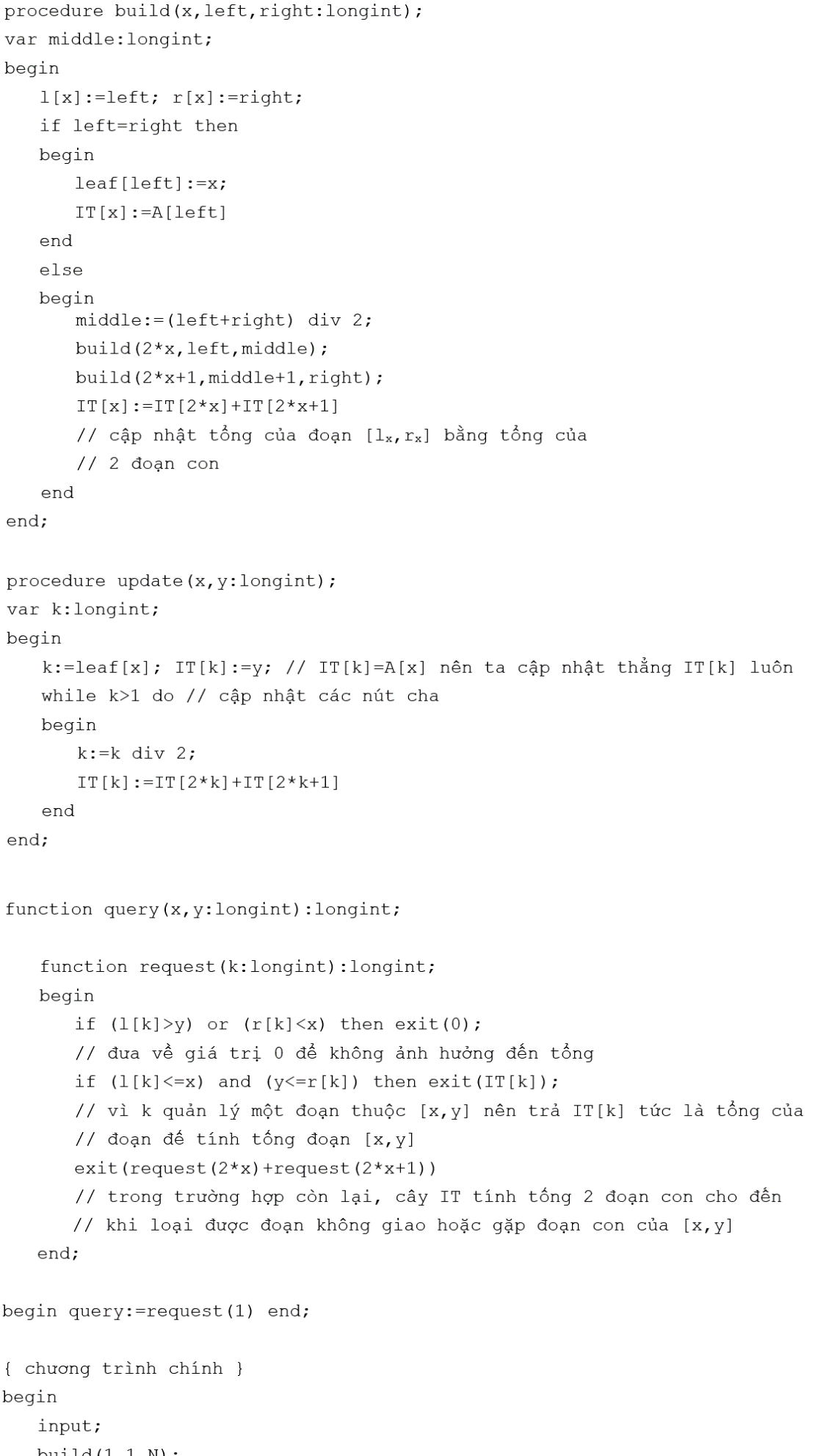
- : Cây IT đầy đủ, ở độ sâu cuối cùng có đúng lá, và các độ sâu thấp hơn không có nút lá nào (và các nút này đều có đúng 2 con). Như vậy: tầng có nút, tầng có nút... Tổng các nút không quá nút.

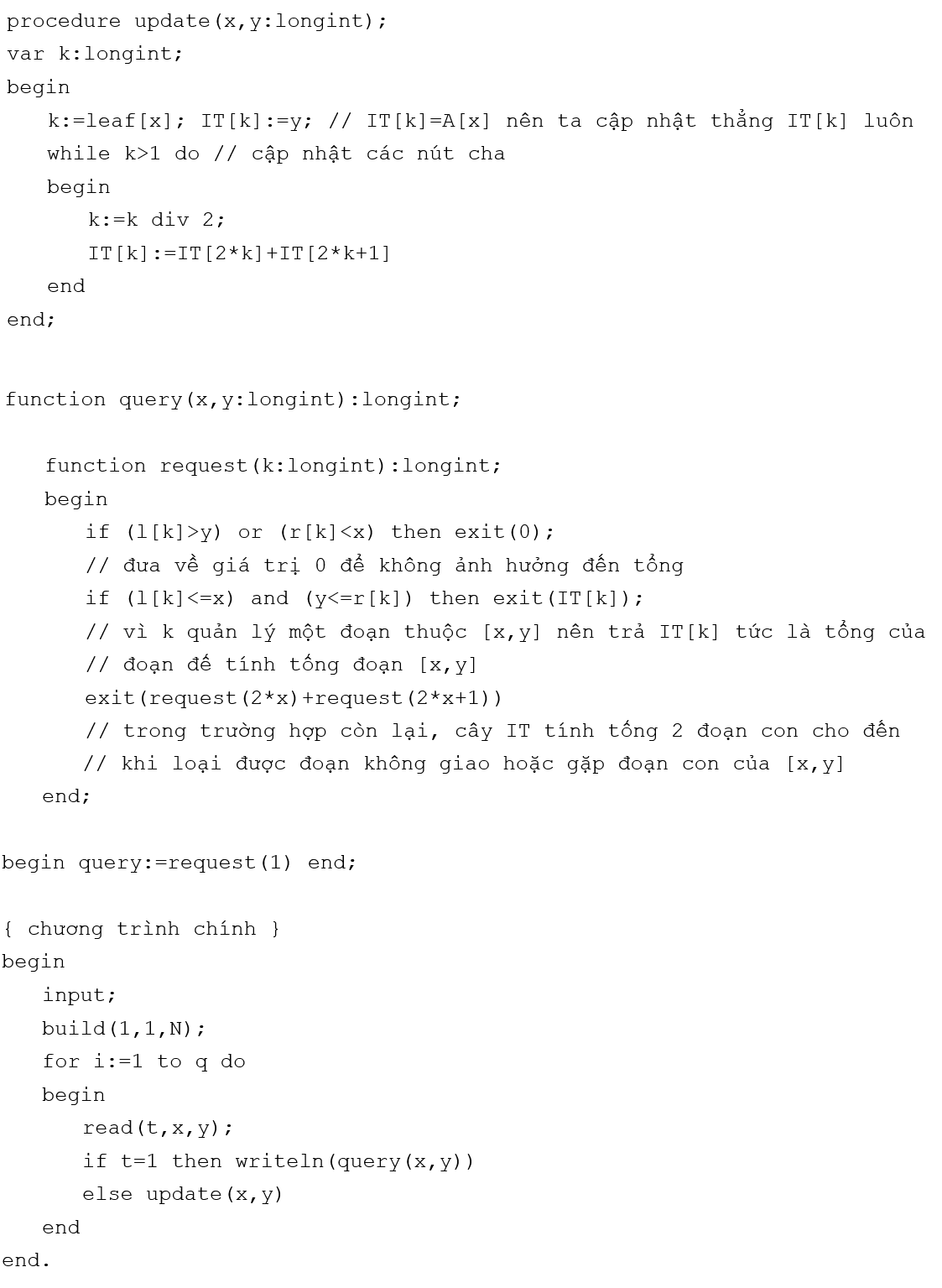
- : Số nút của cây IT chắc chắn sẽ không quá số nút của cây IT có → Tổng các nút không quá nút.

Do đó, số nút của cây cho dãy có phần tử, với , là không quá , giá trị này xấp xỉ . Bằng thực nghiệm cho thấy dùng mảng là đủ.

5. Một số bài tập cơ bản:

*Hướng dẫn:* Ban đầu với mảng ta gọi thủ tục để dựng cây IT với là tổng . Với truy vấn loại 1, ta gọi hàm để đưa ra kết quả. Với truy vấn loại 2, ta gọi để cập nhật lại giá trị của và các nút trên cây IT.





*- vn.spoj.com/problems/NKLINEUP/*

Hướng dẫn: Cây IT mỗi nút lưu 2 giá trị min và max là chiều cao min và chiều cao max trong đoạn quản lý.

Hàm trong trường hợp thì trả về giá trị và để lúc lấy min, max của kết quả không bị ảnh hưởng.

Bài này không có update nên không cần code thủ tục update và cũng không cần khai báo mảng .

Code: <http://ideone.com/C5kRWq>

*- vn.spoj.com/problems/QMAX/*

Hướng dẫn: Đầu tiên cho nguyên mảng bằng 0. Trong đoạn từ đến , mỗi phần tử tăng đơn vị, vậy ta coi đó là đoạn thẳng trải dài từ tọa độ đến tọa độ , sử dụng công thức lùa: để lùa rồi chạy từ 1 đến để tính giá trị của mỗi phần tử.

Sau đó ta dựng cây IT với mỗi nút chứa giá trị max trong đoạn nó quản lý.

Code: <http://ideone.com/a2AaIu>