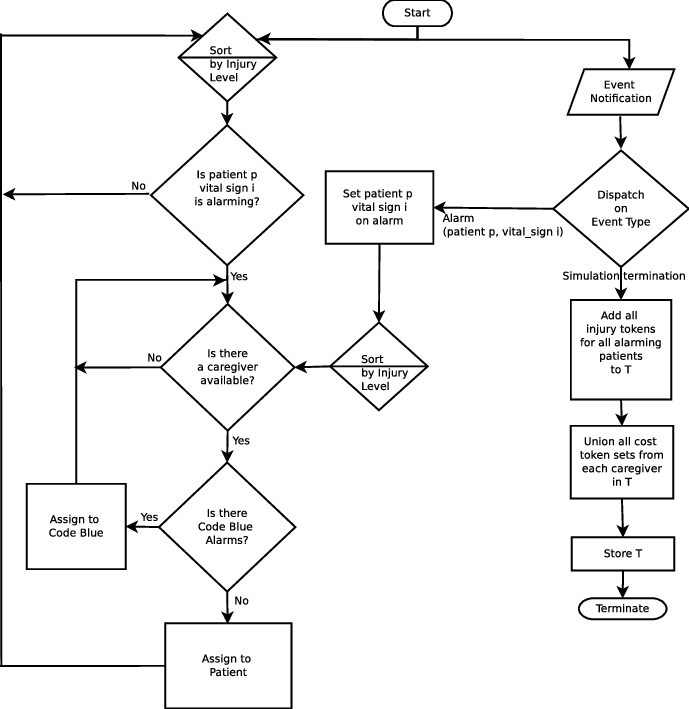
C.Thuật toán được cung cấp trong bài giảng

I. Lí thuyết phương pháp

- Giải thuật tham lam (tiếng Anh: Greedy algorithm) là một thuật toán giải quyết một bài toán theo kiểu metaheuristic để tìm kiếm lựa chọn tối ưu địa phương ở mỗi bước đi với hy vọng tìm được tối ưu toàn cục.

- Lược đồ tổng quát:



- Ưu điểm:

+Dễ triển khai

+Độ phức tạp thời gian thấp

+Có thể được sử dụng để giải quyết bài toán tối ưu cục bộ

-Nhược điểm

+ Không phải lúc nào cũng tìm ra cho bài toán tối ưu toàn cục

II. Lập trình

-Bài toán đổi tiền:

<https://github.com/nanhus/DAA/blob/master/Hw6/CoinChange.java>

Độ phức tạp thời gian: O ( *nW),* trong đó *n* là số loại đồng xu.

-Bài toán xếp ba lô:

<https://github.com/nanhus/DAA/blob/master/Hw6/KnapSack.java>

Độ phức tạp thời gian: là O (2 ^ n), trong đó n là số phần tử có mặt.

III. Bài tập Anany Book

Ex 9.1:

10:

Không cần kiểm tra khả năng kết nối của biểu đồ vì Prim’s algo-rithm có thể tự làm điều đó. Nếu thuật toán đạt đến tất cả các đỉnh của đồ thị (thông qua các cạnh có độ dài hữu hạn), thì đồ thị được kết nối, ngược lại, nó là không

13:

a. Giải pháp đơn giản và hợp lý nhất là gán tất cả trọng số cạnh từ 1

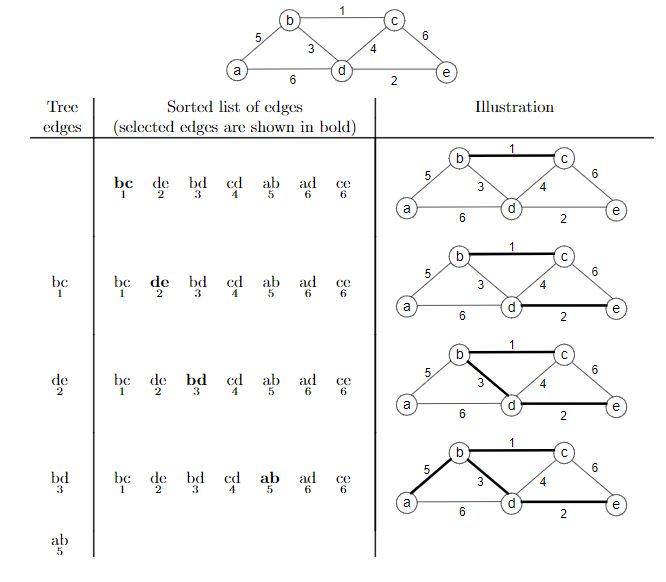
b. Áp dụng tìm kiếm theo chiều sâu (hoặc tìm kiếm theo chiều rộng) theo chiều ngang của cây tìm kiếm theo chiều sâu (hoặc cây tìm kiếm theo chiều rộng), là phương pháp hiển thị khái niệm và cho các biểu đồ thưa thớt được biểu thị bằng danh sách kề của chúng nhanh hơn.

Ex 9.2:

3. Vì số cạnh trong một rừng bao trùm nhỏ nhất của đồ thị có | V | đỉnh và | C | các thành phần liên thông bằng | V | - | C | (for-mula này là một tổng quát hóa đơn giản của | E | = | V | −1 đối với đồ thị liên thông), Kruskal (G) sẽ không bao giờ đi tới | V | −1 ba cạnh trừ khi đồ thị được liên kết. Một cách khắc phục đơn giản là xé vòng lặp while bộ đếm <| V | −1 với while k <| E | để làm cho thuật toán dừng lại sau khi sử dụng hết danh sách đã sắp xếp các cạnh của nó

1:

a:



b:

