Mục Lục

1 Cấu trúc dữ liệu

Mục đích của chương

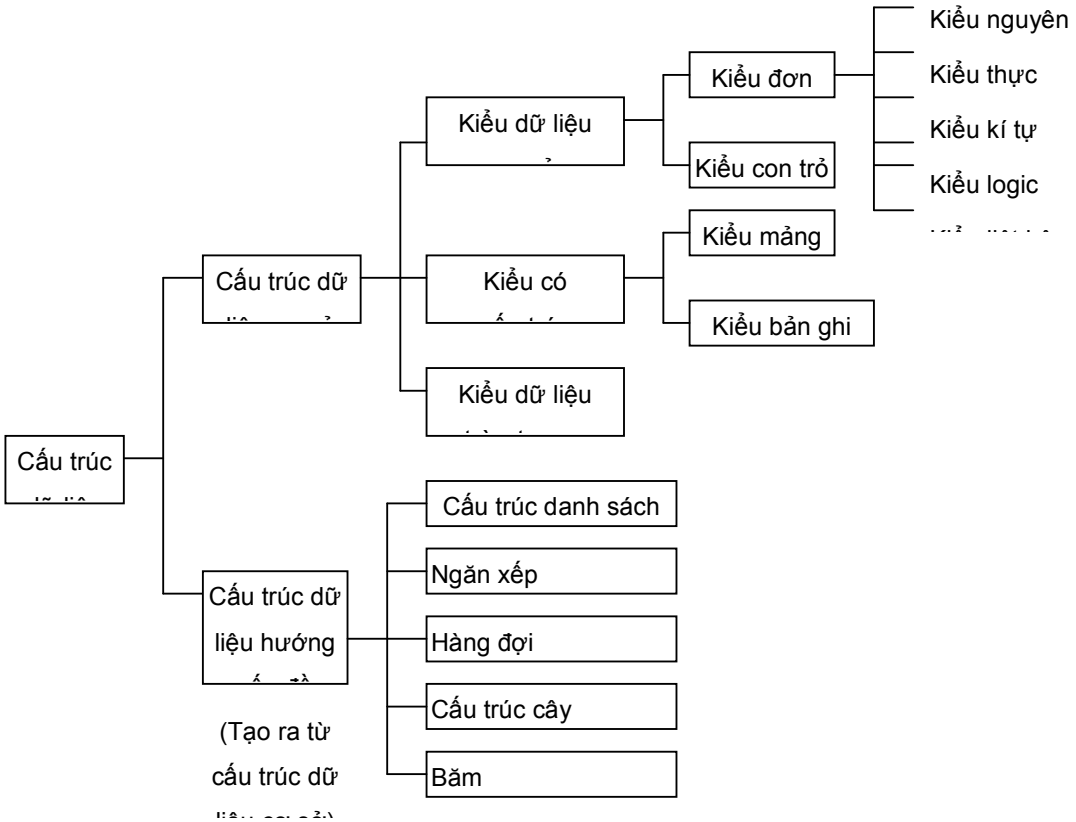
Việc chọn cấu trúc dữ liệu thích hợp nhất và thủ tục mô tả dữ liệu là mấu chốt để tạo ra chương trình hiệu quả, dễ hiểu.
Chương này mô tả các cấu trúc dữ liệu đa dạng bạn cần nắm được xem như bước đầu tiên để học lập trình.

① Hiểu cách phân loại các cấu trúc dữ liệu đa dạng
② Hiểu các kiểu dữ liệu cơ sở thông dụng nhất và mảng dữ liệu
③ Hiểu các đặc trưng và cơ chế của cấu trúc dữ liệu hướng vấn đề được dùng để giải quyết các bài toán đặc biệt, cũng như cách dùng cấu trúc dữ liệu cơ sở cho việc cài đặt chương trình

1.1 Cấu trúc dữ liệu là gì?

Tập các dữ liệu cùng một loại được máy tính xử lí được gọi là "kiểu dữ liệu." Trong giai đoạn thiết kế chương trình, cách thức dữ liệu nên được biểu diễn và lập trình trong máy tính phải được xem xét cẩn thận, để có thể chọn được kiểu dữ liệu thích hợp nhất. Một kiểu dữ liệu được biểu diễn và lập trình được gọi là "cấu trúc dữ liệu."
Hình 1-1-1 chỉ ra phân lớp về các cấu trúc dữ liệu.

Hình 1-1-1 Phân lớp về các cấu trúc dữ liệu



Cấu trúc dữ liệu cơ sở có thể được biểu diễn trong hầu hết tất cả các ngôn ngữ lập trình. Cấu trúc dữ liệu hướng vấn đề là cấu trúc dữ liệu có thể được dùng một cách có hiệu quả để giải quyết những vấn đề chuyên dụng. Có một số cấu trúc dữ liệu hướng vấn đề mà không thể được biểu diễn trong ngôn ngữ lập trình. Trong trường hợp đó, cấu trúc dữ liệu cơ sở được dùng.

1.2 Cấu trúc dữ liệu cơ sở

1.2.1 Kiểu dữ liệu cơ sở

Kiểu dữ liệu cơ sở là tập các dữ liệu riêng lẻ và thường được dùng để tạo ra chương trình. Nó được phân loại thành các kiểu đơn và con trỏ.

(1) Kiểu đơn

Kiểu đơn là kiểu dữ liệu cơ sở nhất. Khi dùng kiểu đơn cho lập trình, kiểu dữ liệu thường được khai báo theo qui tắc cú pháp của ngôn ngữ.

① Kiểu nguyên

Kiểu nguyên biểu diễn cho số nguyên, và được biểu diễn bên trong máy tính như số nhị phân theo số dấu phảy tĩnh, không có chữ số có nghĩa sau dấu chấm thập phân. Giá trị tối đa hay tối thiểu của kiểu nguyên là đơn vị của dữ liệu mà máy tính có thể xử lí vào một lúc, và nó được xác định bởi chiều dài từ.

② Kiểu số thực

Kiểu số thực biểu diễn cho số thực. Nó được dùng để biểu diễn cho số dấu phẩy tĩnh và dấu phẩy động.

③ Kiểu kí tự

Kiểu kí tự biểu diễn cho chữ cái, số và các kí hiệu như các kí tự. Một mã kí tự được biểu diễn như số nhị phân trong máy tính.

④ Kiểu logic

Kiểu logic được dùng để thực hiện các phép toán logic như các phép toán AND, OR và NOT.

⑤ Kiểu liệt kê

Kiểu liệt kê được định nghĩa như kiểu dữ liệu kê ra tất cả các giá trị có thể của biến. Trong trường hợp kiểu liệt kê, có thể kể tên kiểu số nguyên.

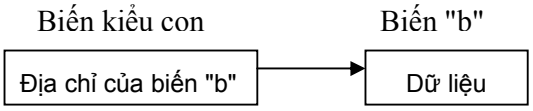
⑥ Kiểu bộ phận

Kiểu bộ phận được dùng để xác định một tập con các giá trị nguyên thuỷ bằng cách hạn chế các kiểu dữ liệu hiện có. Kiểu dữ liệu có các giới hạn trên và dưới như các ràng buộc được gọi là kiểu miền bộ phận.

(2) Kiểu con trỏ

Kiểu con trỏ có địa chỉ được cấp trong đơn vị bộ nhớ chính. Nó được dùng đề tham chiếu tới các biến, các bản ghi tệp hay các hàm. Nó được dùng cho Pascal và C nhưng không dùng cho FORTRAN và COBOL.

Hình 1-2-1 Hình ảnh về kiểu con trỏ



1.2.2 Kiểu có cấu trúc

Cấu trúc dữ liệu có chứa một cấu trúc dữ liệu cơ sở hay bất kì kiểu dữ liệu được xác định nào như phần tử của nó (dữ liệu), được gọi là kiểu có cấu trúc. Kiểu có cấu trúc được phân loại thành kiểu mảng và kiểu bản ghi.

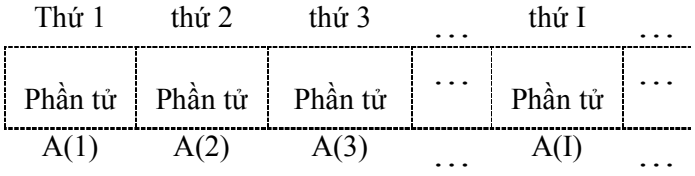
(1) Kiểu mảng

Mảng được gọi là bảng. Kiểu mảng là dữ liệu có cấu trúc có chứa dữ liệu thuộc cùng kiểu và kích cỡ. Từng dữ liệu cá nhân được gọi là một phần tử mảng, phần tử bảng hay phần tử. Cách mảng được mô tả hoặc cách dữ liệu được bố trí có thay đổi tuỳ theo ngôn ngữ lập trình được dùng.

① Mảng một chiều

Mảng một chiều có cấu trúc dữ liệu mà dữ liệu được sắp thành mảng theo một hàng. Để xác định một phần tử trong mảng này, trước hết đưa vào dấu ngoặc tròn mở (hay dấu ngoặc vuông [sau tên của mảng, rồi đưa vào chỉ số và dấu ngoặc tròn đóng) hay dấu ngoặc vuông đóng]. Chỉ số chỉ ra số thứ tự tính từ đỉnh của mảng, nơi phần tử xác định đó được định vị. Mảng "A" có số phần tử được kí hiệu là "i" được biểu diễn là A (i).

Hình 1-2-2 Mảng một chiều



② Mảng hai chiều

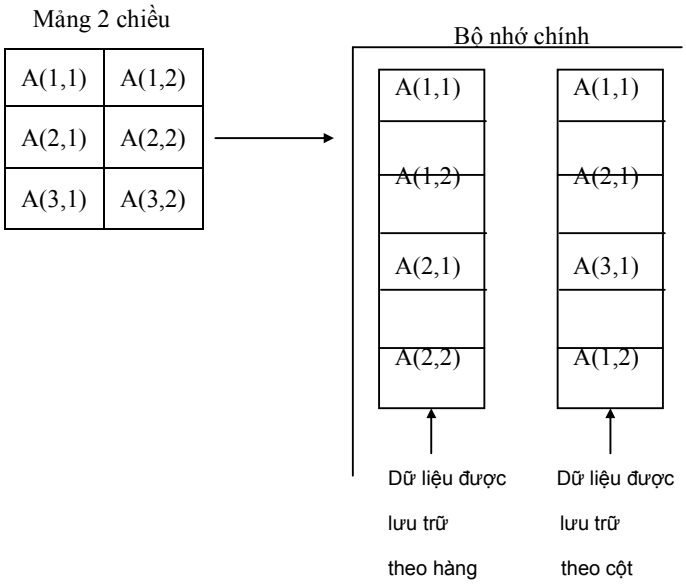
Một cấu trúc dữ liệu trong đó dữ liệu được sắp hàng theo cả hai chiều ngang và đứng được gọi là mảng hai chiều. Dữ liệu theo chiều đứng được gọi là cột và dữ liệu theo chiều ngang được gọi là hàng. Để xác định phần tử nào đó trong mảng này, hai chỉ số trở nên cần thiết: một chỉ số thứ tự theo chiều đứng (trên hàng nào) nơi phần tử xác định đó được định vị và chỉ số kia chỉ ra số thứ tự nào theo chiều ngang (trong cột nào) mà nó được định vị. Chẳng hạn, mảng "A" được định vị ở hàng "i" và cột "j" có thể được diễn tả là A (i, j).

Hình 1-2-3 Mảng hai chiều (với ba hàng và hai cột)

	Cột 1	
Hàng 1	A(1, 1)	A(1, 2)
	A(2, 1)	A(2, 2)
	A(3, 1)	A(3, 2)

Khi mảng hai chiều được lưu giữ trong đơn vị bộ nhớ chính, nó lấy dạng của mảng một chiều. Mảng hai chiều được vẽ trong Hình 1-2-3 lấy dạng của mảng một chiều có sáu phần tử. Như được vẽ trong Hình 1-2-4, dữ liệu được lưu giữ theo kiểu tuần tự hoặc theo chiều của hàng hoặc theo chiều của cột. Chiều theo đó dữ liệu được lưu giữ thay đổi tùy theo trình biên dịch của ngôn ngữ lập trình được dùng. Nói chung, dữ liệu được lưu giữ theo chiều đứng khi Fortran được dùng và theo chiều ngang khi COBOL được dùng.

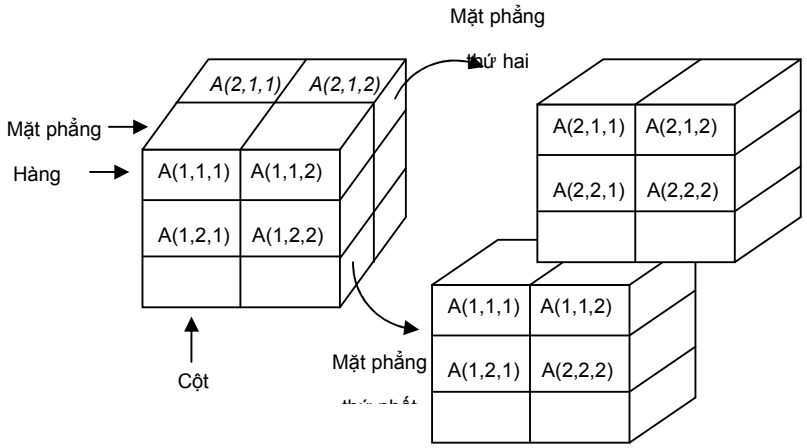
Hình 1-2-4 Cách dữ liệu của mảng hai chiều được lưu giữ trong đơn vị bộ nhớ chính



③ Mảng ba chiều

Mảng ba chiều có cấu trúc dữ liệu nhiều hơn mảng hai chiều. Nó có cấu trúc ba chiều chứa các mặt phẳng, các hàng và cột cũng như các phần tử. Bằng việc xây dựng mảng ba chiều trong mảng hai chiều, có thể xử lí mảng ba chiều theo cùng cách như mảng hai chiều.

Hình 1-2-5 Xây dựng mảng ba chiều thành mảng hai chiều



Mảng nhiều chiều như các mảng bốn, năm hay nhiều chiều cũng có thể được định nghĩa. Tuy nhiên, có thể có những giới hạn nào đó về số chiều, tùy theo kiểu của ngôn ngữ lập trình hay trình biên dịch.

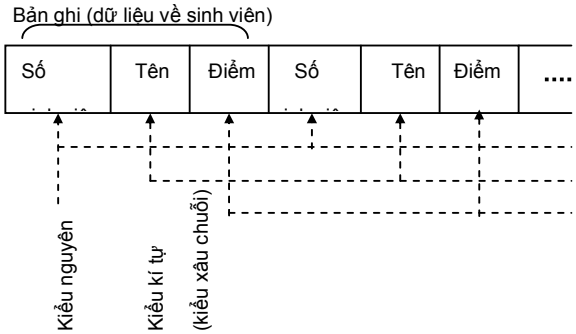
Mảng có thể được phân loại thành mảng tĩnh và mảng động theo phương pháp được dùng để siết chặt một miền.

- Mảng tĩnh: Mảng mà vùng được yêu cầu do chương trình xác định
- Mảng động: Mảng mà vùng được yêu cầu sẽ được xác định ra sau khi chỉ số được dùng cho việc tạo mảng được cung cấp qua một biểu thức và biểu thức đó được tính trong khi thực hiện chương trình

(2) Kiểu bản ghi

Mặc dầu dữ liệu kiểu có cấu trúc là cao cấp hơn trong việc dễ tham chiếu và thực hiện thao tác trên các phần tử, nó cũng có nhược điểm ở chỗ nó chỉ có thể giải quyết dữ liệu thuộc cùng một kiểu. Do đó, dữ liệu có chứa các dữ liệu với kiểu khác nhau phải lấy dạng của dữ liệu kiểu bản ghi. Kiểu bản ghi này cũng còn được gọi là kiểu cấu trúc.

Hình 1-2-6 Kiểu bản ghi



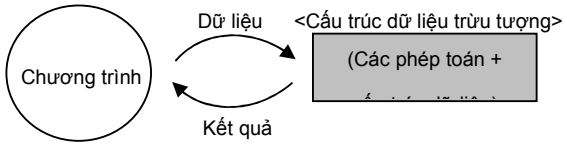
Hình 1-2-6 chỉ ra cấu trúc dữ liệu của kiểu bản ghi. Một bản ghi chứa số hiệu sinh viên (kiểu nguyên), tên (kiểu kí tự) và điểm (kiểu nguyên). Một dữ liệu kiểu bản ghi chứa một tập các bản ghi có cùng định dạng này. Mặc dầu dữ liệu kiểu bản ghi một chiều có thể được

giải quyết theo cùng cách như mảng một chiều, từng dữ liệu vẫn phải được đặt tên để nhận diện vì từng phần tử chứa nhiều dữ liệu.

1.2.3 Kiểu dữ liệu trừu tượng

Dữ liệu chứa cấu trúc dữ liệu nào đó và kiểu của các phép toán được gọi là kiểu dữ liệu trừu tượng. Để truy nhập vào kiểu dữ liệu này, bạn không cần biết về cấu trúc bên trong của nó. Tất cả các dữ liệu đều được che dấu ngoại trừ dữ liệu bạn truy nhập để tham chiếu, thêm vào hay xoá đi. Điều này được gọi là che giấu thông tin. Che giấu thông tin hoặc che giấu dữ liệu ở mức độ kiểu dữ liệu được gọi là bao bọc dữ liệu.

Hình 1-2-7 Kiểu dữ liệu trừu tượng



1.3 Cấu trúc dữ liệu hướng vấn đề

Các cấu trúc dữ liệu hướng vấn đề khác nhau có thể được trù tính bằng việc dùng các kiểu mảng, kiểu con trỏ và các cấu trúc dữ liệu cơ sở khác.

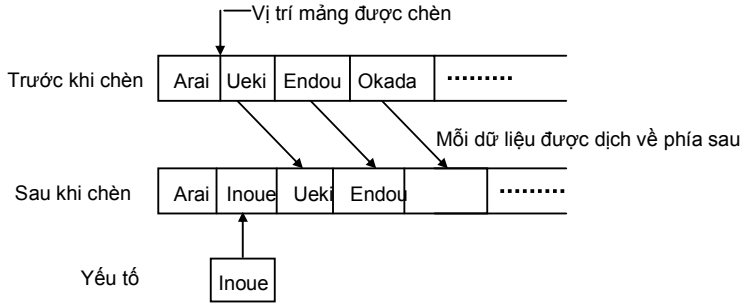
1.3.1 Cấu trúc danh sách

Không giống kiểu dữ liệu cơ sở giải quyết cho từng dữ liệu riêng lẻ, cấu trúc danh sách cho phép dữ liệu được móc nối lẫn nhau và giải quyết cả một cục. Dữ liệu được bố trí theo cấu trúc danh sách này được gọi là một danh sách.

(1) Cấu trúc danh sách và các ô

Bằng việc dùng chỉ số cho từng phần tử trong mảng, có thể truy nhập nhanh chóng vào bất kì phần tử nào. Tương tự như vậy, việc thay đổi dữ liệu có thể được thực hiện dễ dàng. Nếu bạn chèn một dữ liệu vào đâu đó trong mảng, bạn phải dịch chuyển toàn bộ từng dữ liệu sau đó lùi lại một vị trí. Nếu bạn xoá một dữ liệu trong mảng, tương tự, bạn phải dịch chuyển toàn bộ từng dữ liệu sau dữ liệu bị xoá đó nhích lên một vị trí.

Hình 1-3-1 Chèn thêm một phần tử mảng

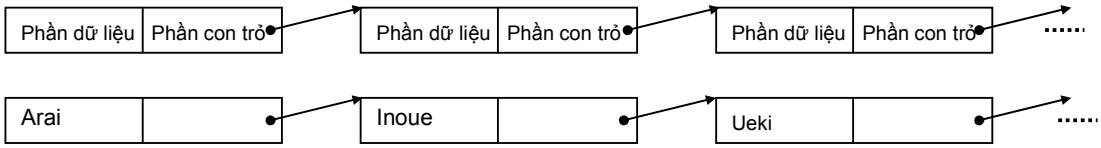


Không giống như cấu trúc kiểu mảng, cấu trúc danh sách cho phép phần tử dữ liệu của cùng kiểu được sắp hàng tuần tự. Kiểm mảng đòi hỏi rằng việc bố trí logic cho các phần tử là giống hệt như việc bố trí vật lí của chúng trong bộ nhớ chính. Trong trường hợp của cấu trúc danh sách, việc bố trí logic không sánh hệt như việc bố trí vật lí.
Danh sách chứa các ô và mỗi ô bao gồm những phần tử sau:

- Phần dữ liệu chứa phần tử dữ liệu
- Phần con trỏ chứa địa chỉ

Do đó, phần dữ liệu của ô có cùng cấu trúc dữ liệu như cấu trúc dữ liệu của dữ liệu được lưu giữ và phần con trỏ của ô có cấu trúc dữ liệu kiểu con trỏ. Điều này nghĩa là các ô biểu diễn cho dữ liệu (cấu trúc) kiểu bản ghi chứa các phần tử có cấu trúc dữ liệu khác nhau. Danh sách chứa địa chỉ ô trong phần con trỏ và ô này được móc nối sang ô kia qua con trỏ.

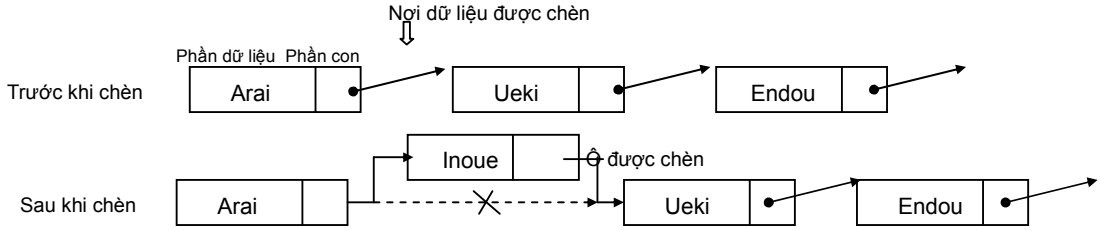
Hình 1-3-2 Cấu trúc danh sách



(2) Chèn dữ liệu vào danh sách

Để chèn dữ liệu vào danh sách, mọi điều bạn cần làm là thay thế các con trỏ tới dữ liệu đi trước và đi sau dữ liệu được chèn vào đó. Bởi vì bạn không phải dịch chuyển các phần tử như trường hợp dữ liệu kiểu mảng, nên bạn có thể chèn thêm dữ liệu một cách dễ dàng và nhanh chóng. (Xem Hình 1-3-3.)

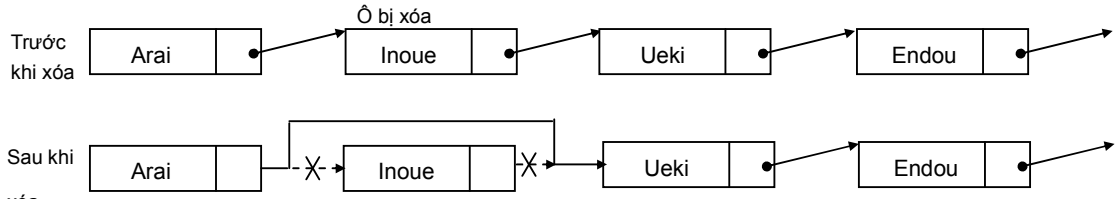
Hình 1-3-3 Chèn dữ liệu vào danh sách



(3) Xoá dữ liệu khỏi danh sách

Để xoá dữ liệu khỏi danh sách, mọi điều bạn cần phải làm là thay thế các con trỏ như khi bạn chèn dữ liệu vào danh sách. Ô chứa dữ liệu (Inoue) vẫn còn trong vùng bộ nhớ như rác sau khi dữ liệu đã bị xoá, như được vẽ trong Hình 1-3-4.

Hình 1-3-4 Xoá dữ liệu khỏi danh sách



Mặc dầu cấu trúc danh sách cho phép dữ liệu được chèn thêm hay được xoá đi chỉ bằng cách thay thế các con trỏ, nó có nhược điểm là bạn phải lần theo từng con trỏ một từ đầu nếu bạn muốn truy nhập vào dữ liệu đặc biệt.

(4) Kiểu cấu trúc danh sách

Các cấu trúc danh sách tiêu biểu bao gồm:

- Danh sách một chiều
- Danh sách hai chiều
- Danh sách vòng.

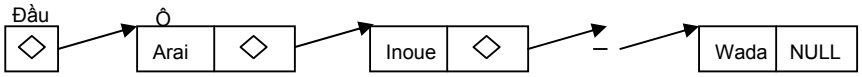
Bởi vì những danh sách này được biểu diễn dưới dạng tuyến thẳng, nên nói chung chúng được gọi là danh sách tuyến tính.

① Danh sách một chiều

Danh sách một chiều cũng còn được gọi là danh sách một hướng. Phần con trỏ của ô chứa

địa chỉ của ô mà trong đó dữ liệu tiếp được lưu giữ. Bằng việc lần theo những địa chỉ này từng ô một, bạn có thể thực hiện việc duyệt dữ liệu.

Hình 1-3-5 Danh sách một chiều

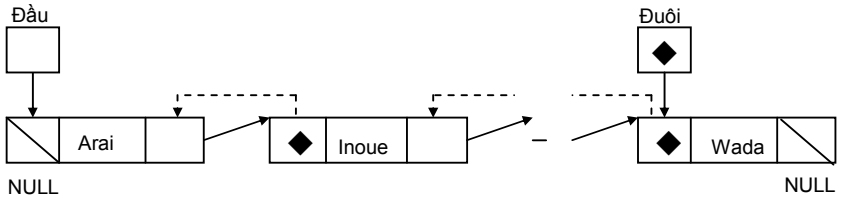


Con trỏ thứ nhất được gọi là gốc hay đầu. Bởi vì phần con trỏ của ô cuối không có bất kì địa chỉ nào trong đó dữ liệu có thể được lưu giữ, nên NULL (giá trị số là không) hay ＼0 được chèn thêm vào phần này.

② Danh sách hai chiều

Danh sách hai chiều có hai phần con trỏ (◇ và ◆) chứa địa chỉ các ô như được vẽ trong Hình 1-3-6.

Hình 1-3-6 Danh sách hai chiều

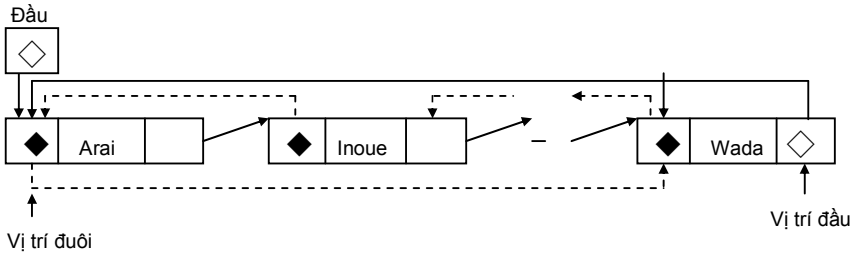


Phần con trỏ ◇ và ◆được vẽ trong Hình 1-3-6 chứa địa chỉ của ô kế tiếp và địa chỉ của ô đứng trước tương ứng. Địa chỉ của ô cuối cùng được chứa trong con trỏ đuôi. Trong trường hợp danh sách hai chiều, dữ liệu có thể được lần theo từ ô đầu hoặc đuôi.

③ Danh sách vòng

Danh sách hai chiều chứa NULL ở ô đầu tiên được gọi là danh sách vòng. Phần con trỏ của ô thứ nhất này và phần con trỏ chứa NULL của ô cuối cùng chứa địa chỉ ô khác, do vậy dữ liệu có dạng cái vòng. Dữ liệu có thể được duyệt theo cùng cách như trong danh sách hai chiều.

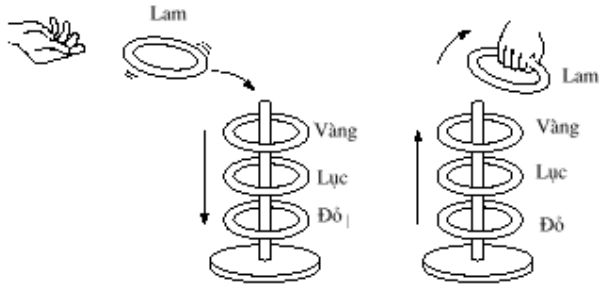
Hình 1-3-7 Danh sách vòng



1.3.2 Ngăn xếp

Ngăn xếp là cấu trúc dữ liệu được thiết kế dựa trên mảng một chiều. Phần tử cuối cùng được lưu giữ sẽ được đọc ra trước hết. Nó được so sánh với trò chơi ném vòng.

Hình 1-3-8 Trò chơi ném vòng

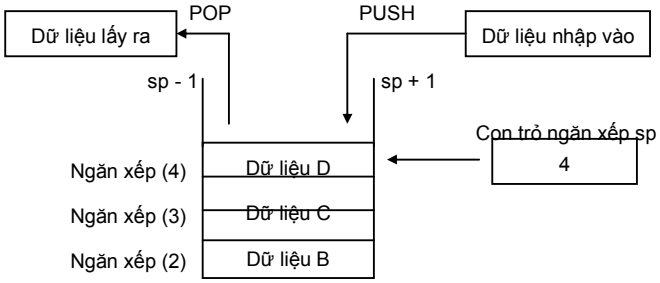


Trò chơi ném vòng được chơi bằng cách ném các vòng mầu theo thứ tự đỏ, lục, vàng và lam (đưa vào dữ liệu). Chúng được lấy ra từng cái một (đưa ra dữ liệu) theo thứ tự đảo lại việc ném vào, tức là lam, vàng, lục và đỏ. Tức là vòng lam được ném vào cuối cùng sẽ được lấy ra đầu tiên.

Kiểu cấu trúc dữ liệu này mà có thể được so sánh với trò chơi ném vòng được gọi là ngăn xếp. Hệ thống này còn có thuật ngữ là hệ thống vào-sau-ra-trước (LIFO). Việc lưu trữ dữ liệu trong ngăn xếp được gọi là "ấn vào (PUSH)" và việc lấy dữ liệu ra từ ngăn xếp được gọi là "bật ra (POP)." Biến điều khiển việc ấn vào và bật ra được gọi là con trỏ ngăn xếp.

Ấn dữ liệu vào ngăn xếp, đặt con trỏ ngăn xếp "sp" là +1 và lưu giữ dữ liệu trong phần tử mảng được viết là "sp." Để làm bật ra dữ liệu từ ngăn xếp, hãy lấy dữ liệu đã được lưu giữ trong mảng được chỉ bởi "sp" và đặt con trỏ ngăn xếp là sp-1.

Hình 1-3-9 Cấu trúc ngăn xếp



1.3.3 Hàng đợi

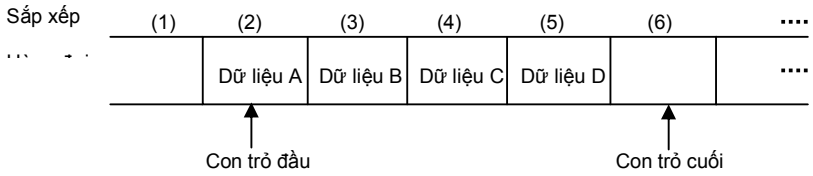
Hàng đợi là cấu trúc dữ liệu dựa trên mảng một chiều. Dữ liệu được lưu giữ đầu tiên được đọc ra đầu tiên. Nó được so sánh với hàng người đang đợi trước máy trả tiền của ngân hàng.

Hình 1-3-10 Hàng đợi

Cấu trúc dữ liệu cho phép khách hàng được phục vụ trên cơ sở đến trước phục vụ trước được gọi là hàng đợi. Hệ thống này được gọi là hệ thống (FIFO). Hai con trỏ chỉ ra đầu và đuôi của hàng đợi là cần cho việc kiểm soát hàng đợi. Con trỏ chỉ ra đầu và đuôi của hàng

đợi được biểu diễn như biến đầu và biến đuôi tương ứng. (Xem Hình 1-3-11.)

Hình 1-3-11 Cấu trúc hàng đợi



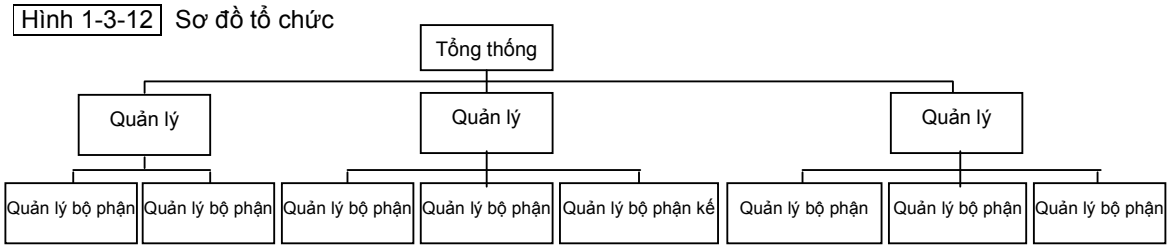
<Thủ tục vận hành hàng đợi>

1. Để lưu giữ dữ liệu vào hàng đợi (enqueue), đặt biến trỏ đuôi tăng thêm 1 và cất giữ dữ liệu.
2. Để lấy ra dữ liệu từ hàng đợi (dequeue), lấy dữ liệu ra và đặt biến trỏ đầu tăng lên 1.

1.3.4 Cấu trúc cây

Cấu trúc cây là một trong những cấu trúc dữ liệu rất hữu dụng vì nó có thể kiểm soát dữ liệu phức tạp tốt hơn các cấu trúc dữ liệu kiểu mảng hay danh sách.
Bởi vì nó có cấu trúc phân cấp như tổ chức của công ti hay cây gia đình, nên nó thích hợp cho kiểu làm việc bao gồm phân loại từng bước.

Hình 1-3-12 Sơ đồ tổ chức

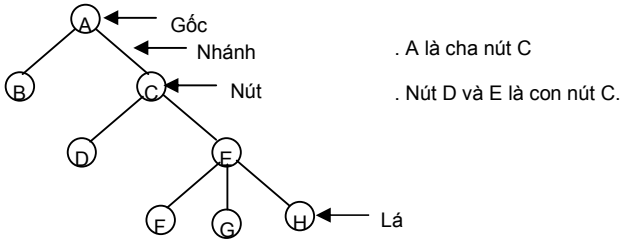


Mặc dầu từng ô được sắp theo thứ tự tuyến tính trong cấu trúc danh sách, dữ liệu được sắp trong khi nó phân nhánh trong cấu trúc cây.
Cấu trúc cây bao gồm các phần tử được vẽ dưới đây:

- Nút: Tương ứng với dữ liệu
- Nhánh: Nối nút này với nút khác
- Gốc: Nút ở cấp cao nhất, không có cha mẹ
- Con: Nút rẽ nhánh ra dưới một nút khác
- Cha mẹ: Dữ liệu gốc trước khi nó chia nhánh
- Lá: Nút ở cấp thấp nhất không có con

Hình 1-3-13 vẽ ra cấu trúc cây

Hình 1-3-13 Cấu trúc cây



(1) Cây nhị phân

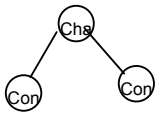
Nếu số nhánh chẽ ra từ một nút là "n" hay ít hơn, tức là nếu số con là 0 cho tới "n", một cấu trúc cây như vậy được gọi là cây N ngôi. Cây N-ngôi có 2 nhánh (n=2), tức là cây N ngôi không có con, có một hay hai con, được gọi là cây nhị phân, thường là cấu trúc dữ liệu thường dùng nhất. Mặt khác, cấu trúc cây có ba hay nhiều nhánh (n>2) được gọi là cây nhiều nhánh.
Cây nhị phân bao gồm một phần dữ liệu và hai phần con trỏ. Con trỏ trái chỉ ra vị trí của nút kéo dài sang bên trái và các nhánh chẽ ra trong khi phần con trỏ phải chỉ ra vị trí của nút kéo dài sang bên phải và các nhánh chẽ ra.

Hình 1-3-14 Cấu trúc cây nhị phân

Phần con trỏ trái	Phần dữ liệu	Phần con trỏ phải

Cây nhị phân có cấu trúc phân cấp cha mẹ - con, như được vẽ trong Hình 1-3-15.

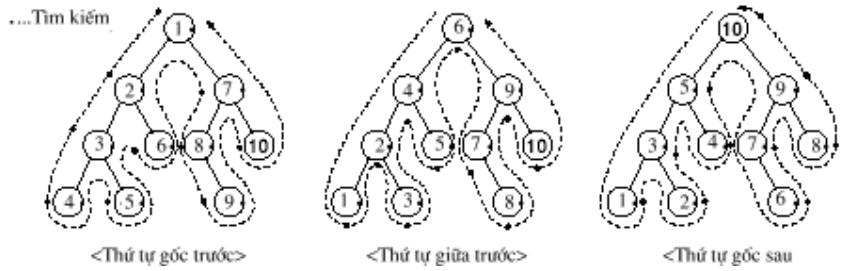
Hình 1-3-15 Cấu trúc cây nhị phân cơ sở



<Cách thực hiện việc duyệt dữ liệu cây nhị phân>

Để duyệt dữ liệu đặc biệt trong dữ liệu cây nhị phân, phải lần theo từng nút một. Có ba phương pháp thực hiện duyệt dữ liệu cây nhị phân. Vì khó giải thích bằng lời nên hãy kiểm lại Hình 1-3-16 và xem cách dữ liệu được duyệt bằng việc dùng từng phương pháp.

Hình 1-3-16 Cách thực hiện duyệt dữ liệu cây nhị phân

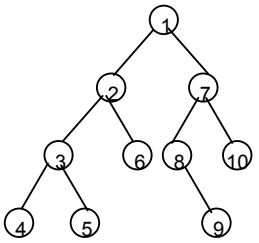


- Thứ tự gốc trước (Pre-order): Với gốc là điểm bắt đầu, nút bên trái của mỗi nút được duyệt qua theo cách tuần tự.
- Thứ tự gốc giữa (Mid-order): Với lá tại đáy bên trái làm điểm bắt đầu, rồi duyệt qua nút cha nó và tiếp đó duyệt qua phần còn lại của nút đó theo cách tuần tự.
- Thứ tự gốc sau (Post order): Với lá tại đáy bên trái làm điểm bắt đầu, phần bên phải mỗi nút được duyệt qua theo cách tuần tự rồi mới đến nút cha của nó.

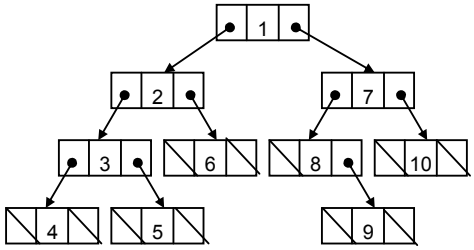
(2) Cây nhị phân hoàn chỉnh

Nếu cây nhị phân được xây dựng theo cách số các nhánh từ gốc tới từng lá dọc theo một nhánh là bằng hoặc sai khác một so với số các nhánh từ gốc tới từng lá dọc theo nhánh khác (hoặc nếu chiều cao từ gốc tới từng lá là bằng hay sai khác một với chiều cao từ gốc tới từng lá thuộc vào nhánh khác), thì cây đó được gọi là cây nhị phân hoàn chỉnh. Hình 1-3-17 vẽ ra cây nhị phân hoàn chỉnh có mười nút.

Hình 1-3-17 Cây nhị phân hoàn chỉnh



<Cây nhị phân hoàn chỉnh>



<Cấu trúc dữ liệu>

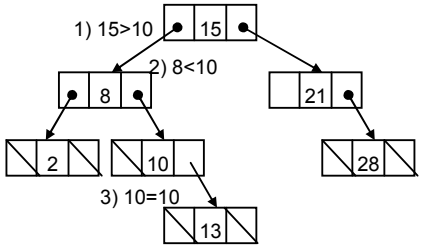
(3) Cây tìm kiếm nhị phân

Cây tìm kiếm nhị phân được dùng như một biến thể của cây nhị phân. Trong trường hợp của cây tìm kiếm nhị phân, con cháu bên trái là nhỏ hơn cha mẹ và con cháu ở bên phải là lớn hơn cha mẹ.

Thuật toán cây tìm kiếm nhị phân là như sau:

1. Gốc là điểm việc tìm kiếm bắt đầu.
2. Dữ liệu cây nhị phân được so sánh với dữ liệu cần tìm.
3. Nếu dữ liệu cây nhị phân = dữ liệu cần tìm, việc tìm là thành công (được hoàn tất).
4. Nếu dữ liệu cây nhị phân > dữ liệu cần tìm, thì các nút bên trái của cây nhị phân được tìm và so sánh.
5. Nếu dữ liệu cây nhị phân < dữ liệu cần tìm, thì các nút bên phải của cây nhị phân được tìm và so sánh.
6. Nếu không tìm thấy con nào, việc tìm kiếm là không thành công (không tìm thấy dữ liệu).

Hình 1-3-18 Thực hiện việc tìm kiếm về dữ liệu trong cây tìm kiếm nhị phân



(4) Cây cân bằng

Nếu dữ liệu được thêm vào hay bị xoá đi trong cấu trúc cây, thì các lá ngẫu nhiên phát triển và tính hiệu quả của phép toán sẽ giảm đi. Cấu trúc cây có khả năng tự tổ chức lại chính nó được gọi là cây cân bằng. Cây cân bằng đại diện là B-cây và đống (heap).

① B-cây

B-cây là phiên bản được phát triển thêm nữa của cây nhị phân:

- Lá cuối bị bỏ đi và mỗi nút đều có số các nút được xác định là "m."
- Mỗi nút đều có số tối đa dữ liệu được xác định là "m-1."
- Tất cả các lá đều trên cùng một mức.
- Dữ liệu được chứa trong từng nút được bố trí trong hàng đợi.

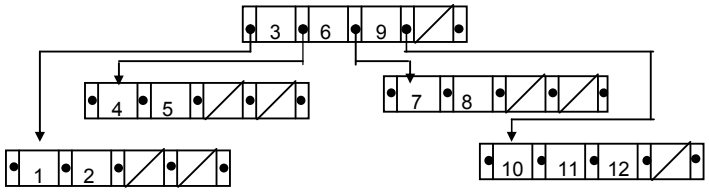
Bởi vì những đặc trưng trên có thể làm tăng bộ nhớ và tính hiệu quả tính toán, nên tên "B-cây" nghĩa là cây cân bằng tốt.

a. Đặc trưng của B-cây

- Để tăng việc sử dụng vùng bộ nhớ, số con trỏ mà mỗi nút có được đặt là m/2 hoặc nhiều hơn và là m hoặc ít hơn. Số con trỏ theo một đường tuy vậy được đặt là 2 hoặc nhiều hơn.
- Mỗi lúc dữ liệu bị xoá hay được bổ sung thêm, việc chẻ ra và ghép lại được thực hiện tự động để cho số các phần tử của một nút có thể trở thành m/2 hay nhiều hơn hoặc m hay ít hơn. Điều này cho phép lá bao giờ cũng được duy trì trên cùng mức.
- Việc tìm kiếm bắt đầu từ gốc.
- Việc thêm vào hay xoá đi dữ liệu bắt đầu từ lá.

Hình 1-3-19 chỉ ra B-cây bộ năm với các phần tử │7, 6, 10, 2, 5, 12, 4, 9, 8, 11, 1, 3│.

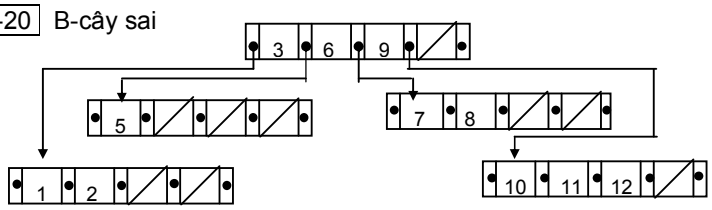
Hình 1-3-19 B-cây bộ năm



b. Xoá dữ liệu

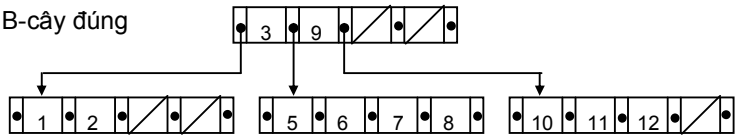
Hình 1-3-20 vẽ ra trường hợp dữ liệu "4" bị xoá khỏi B-cây được vẽ trong Hình 1-3-19. Nếu một mình "4" bị xoá đi, thì số các phần tử bị xoá của một nút trở thành một và các yêu cầu của B-cây không thể được đáp ứng.

Hình 1-3-20 B-cây sai



Nút từ đó "4" bị xoá đi được gộp vào một nút kề hoặc ở bên phải hoặc ở bên trái. Bởi vì các con trỏ của cha mẹ cũng phải được tổ chức lại, nên "6" được chuyển sang nút cấp thấp hơn và từng nút có thể được tổ chức lại, như được vẽ trong Hình 1-3-21.

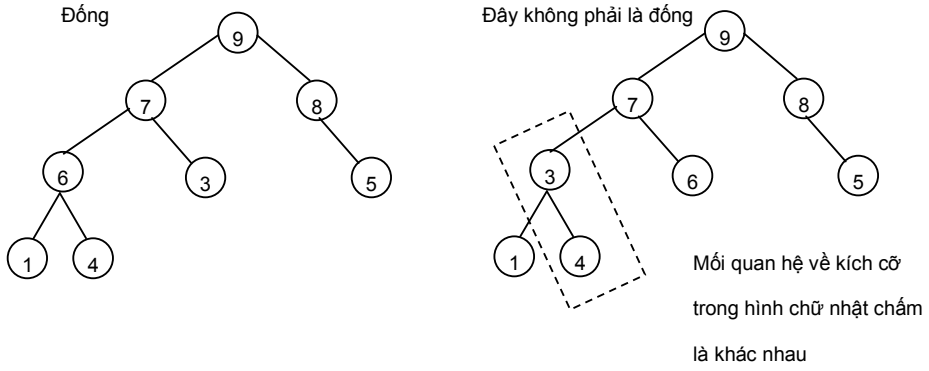
Hình 1-3-21 B-cây đúng



② Đống

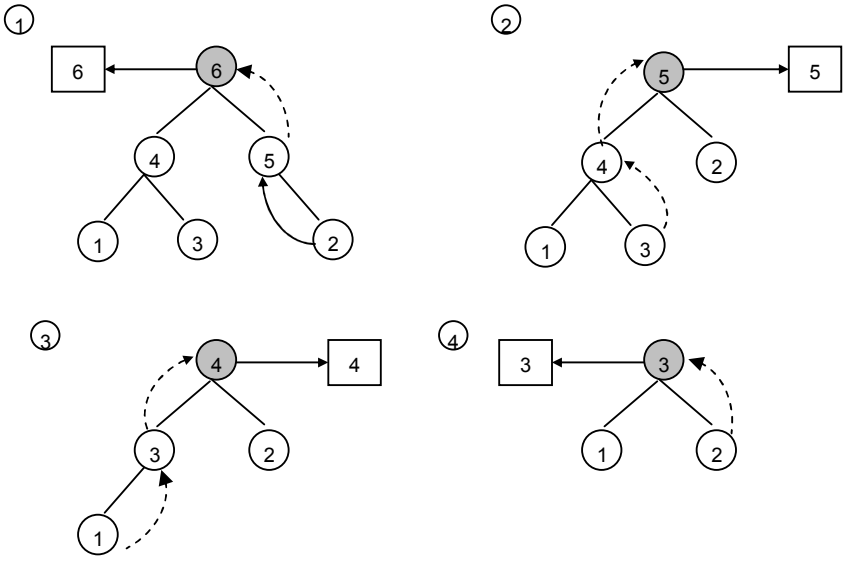
Cây nhị phân hoàn chỉnh có mối quan hệ kích cỡ nào đó giữa nút cha mẹ và nút con được gọi là đống (heap). Đống khác với cây nhị phân ở chỗ đống không có mối quan hệ kích thước nào đó giữa các anh em.

Hình 1-3-22 Ví dụ về đống



Bởi vì đống có cấu trúc của cây nhị phân hoàn chỉnh, nên nó là một loại cây cân bằng, tổ chức được. Trong trường hợp của đống, giá trị tối đa (hay giá trị tối thiểu) của tất cả các dữ liệu được ghi lại trong gốc. Bằng việc dùng đặc trưng này, dữ liệu có thể được sắp xếp bằng việc lấy ra dữ liệu tại gốc theo cách tuần tự.

Hình 1-3-23 Sắp xếp dữ liệu dùng đống



1.3.5 Băm

Băm là cách dùng một cấu trúc dữ liệu kiểu mảng. Với việc dùng băm, bạn có thể truy nhập trực tiếp vào dữ liệu đặc biệt bằng việc dùng một khoá mà không phải truy nhập lần lượt vào dữ liệu được ghi.

(1) Chuyển đổi sang địa chỉ băm

Để chuyển một khoá sang địa chỉ băm (chỉ số), người ta dùng một hàm băm. Hàm băm tiêu biểu là:

- Phương pháp chia
- Phương pháp đăng kí
- Phương pháp đổi cơ sở.

① Phương pháp chia

Khoá được chia theo một giá trị nào đó (một số nguyên tố gần nhất với số phần tử mảng thường được dùng) và số dư được dùng làm địa chỉ (chỉ số) của khoá.

Ví dụ Khoá: 1234 và số phần tử mảng : 100

1234÷97 (số nguyên tố gần 100 nhất) = 12 70 : địa chỉ (chỉ số)

② Phương pháp đăng kí

Khoá được phân tách theo một qui tắc nào đó và tổng được dùng làm địa chỉ (chỉ số) của khoá.

Ví dụ Khoá: 1234
1234 được phân tách thành 12 và 34 và cả hai số này được lấy tổng.

12 + 34 = 46 : Địa chỉ (chỉ số)

③ Phương pháp chuyển cơ số

Khoá thường được biểu diễn theo số hệ thập phân, Khoá này được chuyển thành cơ số khác với thập phân (chẳng hạn cơ số ba), và kết quả được dùng làm địa chỉ (chỉ số) của khoá.

Ví dụ Khoá bản ghi: 1234

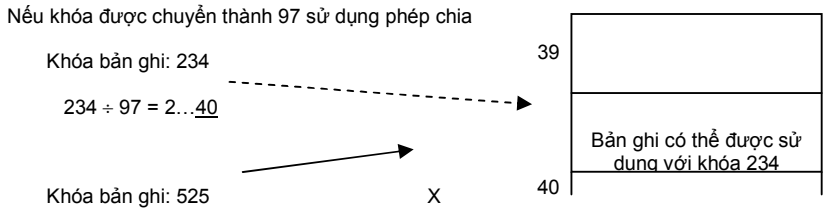
$$1 \times 3^3 + 2 \times 3^2 + 3 \times 3^1 + 4 \times 3^0 = 27 + 18 + 9 + 4$$
$$= 58 \text{ : Địa chỉ (chỉ số)}$$

* Giá trị của từng chữ số theo hệ cơ số 3 là 3 hay nhỏ hơn. Tuy nhiên trong trường hợp này sự kiện này không cần được tính tới.

(2) Đồng nghĩa

Khi một khoá được chuyển thành địa chỉ bằng việc dùng hàm băm, các khoá khác nhau có thể được chuyển vào cùng một địa chỉ. Điều này được gọi là đồng nghĩa (hay đụng độ) (xem Hình 1-3-24). Một bản ghi có thể dùng địa chỉ đã chuyển đổi được gọi là bản ghi nhà còn bản ghi không thể dùng được nó sẽ được gọi là bản ghi đồng nghĩa.

Hình 1-3-24 Xuất hiện đồng nghĩa



Để giải quyết sự đồng nghĩa, một phương pháp đặc biệt hay phương pháp dây chuyền được dùng:

① Phương pháp tuần tự

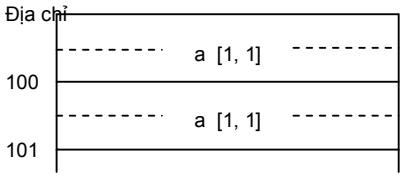
Bằng việc dùng phương pháp tuần tự, bản ghi đồng nghĩa được lưu giữ trong không gian tự do được định vị gần địa chỉ mong muốn. Có khả năng là sự đồng nghĩa lại có thể xuất hiện theo phản ứng dây chuyền.

② Phương pháp dây chuyền

Với việc dùng phương pháp dây chuyền, một vùng bộ nhớ tách biệt được thiết lập và các bản ghi đồng nghĩa được lưu giữ trong vùng này. Trong trường hợp này, cần cung cấp một con trỏ chỉ ra địa chỉ nơi các bản ghi đồng nghĩa được cất giữ.

Bài tập

Q1 **Khi lưu giữ mảng hai chiều "a" có mười hàng và mười cột trong không gian bộ nhớ liên tục theo hàng, địa chỉ nơi lưu giữ [5, 6] là gì? Trong câu hỏi này, địa chỉ được biểu diễn theo số thập phân.**



a. 145 b. 185 c. 190 d. 208 e. 212

Q2 **Hình dưới đây là danh sách một chiều. Tokyo đứng đầu trong danh sách này và con trỏ chứa địa chỉ của dữ liệu được nêu dưới đây. Nagoya đứng cuối của danh sách và con trỏ chứa 0. Hãy chọn một cách đúng để đưa thêm Shizuoka vào địa chỉ 150 giữa Atami và Hamamatsu.**

Con trỏ đầu
10

Địa chỉ	Dữ liệu	Con trỏ
10	Tokyo	50
30	Nagoya	0
50	Shin Yokohama	90
70	Hamamatsu	30
90	Atami	70
150	Shizuoka	

a. Con trỏ cho Shizuoka được đặt là 50 và con trỏ cho Hamamatsu được đặt là 150
b. Con trỏ cho Shizuoka được đặt là 70 và con trỏ cho Atami được đặt là 150
c. Con trỏ cho Shizuoka được đặt là 90 và con trỏ cho Hamamatsu được đặt là 150
d. Con trỏ cho Shizuoka được đặt là 150 và con trỏ cho Atami được đặt là 90

Q3 **Cấu trúc dữ liệu thích hợp cho thao tác FIFO (vào trước ra trước) là gì?**

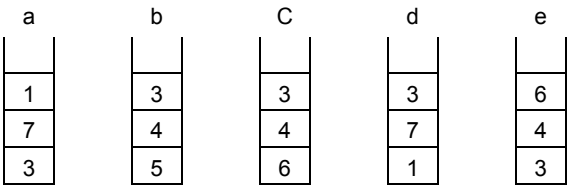
a. Cây nhị phân b. Hàng đợi c. Ngăn xếp d. Đống

Q4 **Hai thao tác ngăn xếp được định nghĩa như sau:**

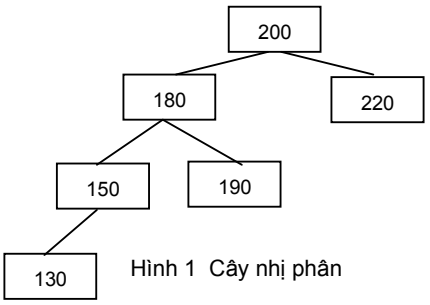
PUSH n: Dữ liệu (nguyên "n") được đẩy vào ngăn xếp.
POP: Dữ liệu được bật ra khỏi ngăn xếp.

Nếu thao tác ngăn xếp được thực hiện trên ngăn xếp rỗng theo thủ tục được nêu dưới đây, thì có thể thu được kết quả gì?

PUSH 1 → PUSH 5 → POP → PUSH 7 → PUSH 6 → PUSH 4 →POP → POP → PUSH 3



Q5 **Hình 2 là biểu diễn mảng cho cây nhị phân được vẽ trong Hình 1. Giá trị nào nên được đặt vào chỗ "a"?**



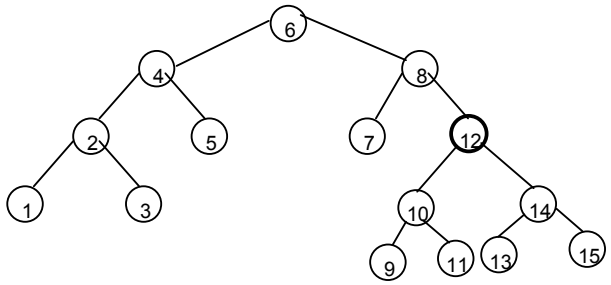
Hình 1 Cây nhị phân

Chỉ số	Giá trị	Con trỏ 1	Con trỏ 2
1	200	3	2
2	220	0	0
3	180	5	a
4	190	0	0
5	150	6	0
6	130	0	0

Hình 2
Biểu diễn mảng cho cây nhị phân

a. 2 b. 3 c. 4 d. 5

Q6 **Khi phần tử 12 bị xoá khỏi cây tìm kiếm nhị phân được vẽ dưới đây, phần tử nào nên được chuyển tới điểm phần tử đã bị xoá để tổ chức lại cây tìm kiếm nhị phân?**

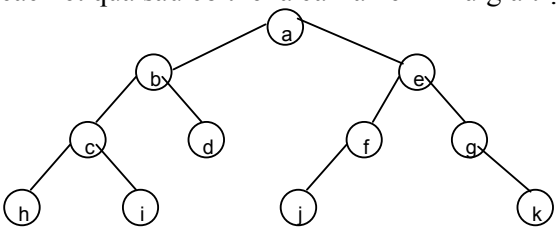


a. 9 b. 10 c. 13 d. 14

Q7 **Nếu hai hay ít hơn hai nhánh chẽ ra từ từng nút của một cây, thì cây như vậy được gọi là cây nhị phân. Cây nhị phân bao gồm một nút, một cây trái và một cây phải. Có ba phương pháp thực hiện tìm kiếm trên cây này:**

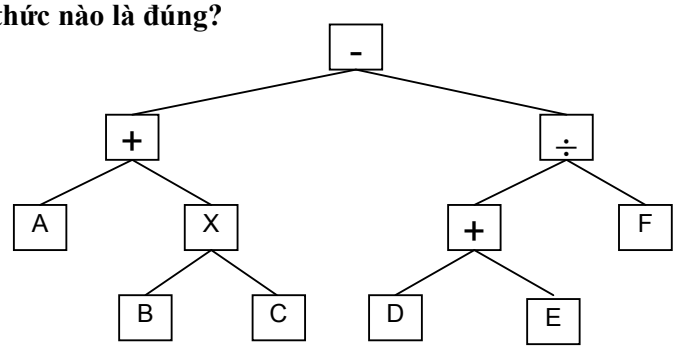
(1) Thứ tự gốc trước: Việc tìm kiếm được thực hiện theo thứ tự của nút, cây trái rồi đến cây phải.
(2) Thứ tự gốc giữa: Việc tìm kiếm được thực hiện theo thứ tự của cây trái, nút và cây phải.
(3) Thứ tự gốc sau: Việc tìm kiếm được thực hiện theo thứ tự cây trái, cây phải và nút.

Nếu việc tìm kiếm được thực hiện bằng việc dùng phương pháp thứ tự gốc sau, thì kết quả nào trong các kết quả sau có thể là cái ra xem như giá trị của nút?



a. abchidefjgk b. abechidfjgk c. hcibdajfegk d. hicdbjfkgea

Q8 **Cây nhị phân được vẽ dưới đây có thể được biểu diễn bằng việc dùng biểu thức số học. Biểu thức nào là đúng?**

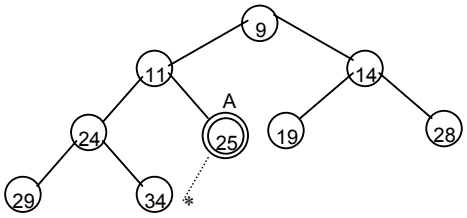


a. A + B × C + (D + E) ÷ F
b. A + B × C - (D + E) ÷ F
c. A + B × C - D + E ÷ F
d. A × B + C + (D - E) ÷ F
e. A × B + C - D + E ÷ F

Q9 **Xét tới việc lưu giữ dữ liệu bằng cách dùng B-cây, hãy chọn câu chú thích đúng từ những câu sau:**

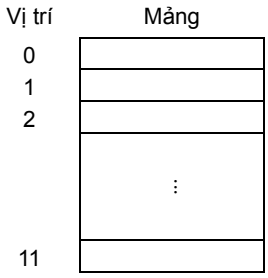
a. Chia ra và gộp lại các nút để cho phép chiều sâu phân cấp trở thành như nhau.
b. Nhận diện vị trí nơi dữ liệu được lưu giữ bằng việc dùng một hàm nào đó và giá trị khoá.
c. Chỉ có thể truy nhập tuần tự tới dữ liệu đầu và dữ liệu tiếp đó.
d. Có danh mục và một thành viên. Thành viên là tệp được tổ chức tuần tự.

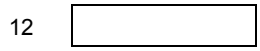
Q10 **Có một đống với giá trị của nút cha mẹ nhỏ hơn giá trị của nút con. Để chèn thêm dữ liệu vào trong đống này, bạn có thể thêm một phần tử vào phần sau nhất và lặp lại việc trao đổi cha mẹ với con cái khi phần tử này nhỏ hơn cha mẹ. Khi phần tử 7 được thêm vào vị trí * của đống tiếp, phần tử nào sẽ vào vị trí A?**



a. 7 b. 9 c. 11 d. 24 e. 25

Q11 **Một số có năm chữ số (a_1 a_2 a_3 a_4 a_5) phải được lưu giữ trong một mảng bằng việc dùng phương pháp băm. Nếu hàm băm được xác định như mod (a_1+a_2+a_3+a_4+a_5, 13) và nếu số có năm chữ số được lưu giữ trong một phần tử mảng ở vị trí sánh với giá trị băm đã được tính, thì tại vị trí đó 54321 sẽ được đặt vào đâu trong mảng được vẽ dưới đây? Ở đây, giá trị mod (x, 13) là phần dư khi lấy x chia cho 13.**





a. 1 b. 2 c. 7 d. 11

Q12 **Năm dữ liệu với các giá trị khoá được phân bố đều và ngẫu nhiên trong phạm vi từ 1 tới 1,000,000 phải được đăng kí vào bảng băm kích cỡ 10. Xác suất xấp xỉ cho đụng độ xảy ra là gì? Ở đây, phần dư thu được khi một giá trị khoá được chia theo kích cỡ của bảng băm được dùng như giá trị băm.**

a. 0.2 b. 0.5 c. 0.7 d. 0.9

2 Thuật toán

Mục đích của chương

Các cơ sở của việc lập trình là thiết kế thuật toán. Trong thiết kế cấu trúc logic của chương trình, điều quan trọng là dùng thuật toán thích hợp nhất. Tương tự như vậy, trong việc chọn chương trình từ thư viện, thuật toán nào được dùng là một trong những tiêu chuẩn quan trọng nhất để chọn chương trình thích hợp nhất.
Chương này mô tả cho các cơ sở của thiết kế thuật toán và các thuật toán tiêu biểu.

① Hiểu các cơ sở của thuật toán, như định nghĩa thuật toán, thiết kế, mối quan hệ với cấu trúc dữ liệu, phương pháp biểu diễn v.v..
② Hiểu đặc trưng và ý nghĩa của các thuật toán tìm kiếm, sắp xếp, xử lí kí tự và xử lí tệp tiêu biểu.

Giới thiệu

Việc dùng thuật toán hiệu quả, dễ hiểu làm cho người ta có khả năng tăng tốc độ thực hiện và giảm số lỗi còn bị giấu kín. Thuật toán là một trong những nhân tố mấu chốt xác định ra hiệu năng hệ thống. Cũng vậy, chất lượng của thuật toán được dùng làm tiêu chuẩn cho việc chọn các bộ phận từ thư viện.
Chương này mô tả các cơ sở của thuật toán được dùng cho thiết kế logic mô đun và những thuật toán được dùng để giải các bài toán điển hình. Việc chọn một thuật toán nổi tiếng hay thường được dùng mà không phân tích đầy đủ các vấn đề được nêu ra là điều cần tránh. Nên chọn lấy một thuật toán thích hợp nhất cho các đặc trưng của bài toán.
Việc đánh giá bản thân thuật toán cũng là một nhiệm vụ quan trọng. Dữ liệu thu được bằng việc so sánh nhiều thuật toán trên cơ sở con số khách quan sẽ giúp ích rất nhiều cho bạn trong việc chọn thuật toán thích hợp nhất.
Trong chương này, các bạn sẽ học những cơ sở về thuật toán để cho các bạn có thể thiết kế mô đun dễ hiểu.

2.1 Cơ sở về thuật toán

Mục này giải thích về:
- Định nghĩa về thuật toán
- Mối quan hệ giữa thuật toán và cấu trúc dữ liệu

2.1.1 Thuật toán là gì?

(1) Định nghĩa về thuật toán

Thuật toán được định nghĩa trong Chuẩn công nghiệp Nhật (JIS) là như sau:
Một tập giới hạn các qui tắc đã được xác định rõ, được áp dụng một số lần để giải quyết các vấn đề.
Điều này có nghĩa là thuật toán là tập các qui tắc (thủ tục) được thiết lập để giải quyết các bài toán. Giải một bài toán, chúng ta có thể lấy nhiều con đường. Xem như một ví dụ đơn giản, khi chúng ta tính một giá trị Y lần lớn hơn X, chúng ta có thể lấy hai cách tiếp cận:
- Nhân X với Y
- Cộng Y lần với X

Trong việc xác định thuật toán, điều quan trọng không chỉ nghĩ về thủ tục giải bài toán mà còn thiết kế và thích ứng thuật toán sao cho có thể giải bài toán một cách có hiệu quả và hiệu lực.
Một điểm quan trọng khác là ở chỗ thuật toán phải có điểm dừng (nó phải không chạy vào chu trình vô hạn). Vấn đề này sẽ được xét tới ở mục 2.3.2 Đánh giá theo tính đúng đắn.

(2) Thuật toán và lập trình

Thuật toán và lập trình là hai mặt của cùng đồng tiền. Lập trình là việc mô tả dữ liệu và thuật toán trong các ngôn ngữ lập trình sao cho máy tính có thể thực hiện các nhiệm vụ được giao của nó.

Nói chung, chương trình bao gồm các thuật toán và dữ liệu còn thuật toán bao gồm logic và điều khiển.
Lập trình có thể được phân loại thành bốn kiểu khác nhau tương ứng với cách thuật toán được xem xét:

- Lập trình thủ tục
- Lập trình hàm
- Lập trình logic
- Lập trình hướng đối tượng

Kiểu lập trình thích hợp nhất phải được chọn có xem xét tới các đặc trưng của bài toán.

① Lập trình thủ tục

Lập trình thủ tục là kiểu lập trình thường được dùng nhất. Loại lập trình này bao gồm các ngôn ngữ lập trình sau, FORTRAN, COBOL, PL/I, Pascal, C, v.v..
Đặc trưng

- Chương trình được chia thành các mô đun để làm cho chương trình phức tạp, lớn thành dễ hiểu. Việc viết mã được thực hiện cho từng mô đun.
- Các định lí có cấu trúc được đưa vào lập trình (sẽ được giải thích về sau).
- Tính năng có cấu trúc được dùng để dịch chương trình

Bởi vì lập trình thủ tục là hướng (điều khiển) thủ tục, nên có những hạn chế sau:

- Bên cạnh thủ tục (điều khiển), các biến (tên biến, kiểu, kích cỡ v.v..) phải được khai báo.
- Các lệnh được thực hiện từng lệnh một theo cách tuần tự (xử lí song song không thể được thực hiện).
- Việc so sánh và tính toán phải được thực hiện để giải bài toán.

② Lập trình hàm

Lập trình hàm là việc lập trình theo hướng dùng các hàm. Nó được dùng trong lĩnh vực trí tuệ nhân tạo (AI), các lí thuyết tính toán cơ sở và các nhiệm vụ nghiên cứu khác. LISP, trong số các ngôn ngữ khác, là ngôn ngữ lập trình hàm.
Đặc trưng

- Không giống lập trình kiểu thực hiện tuần tự, các biểu thức được xây dựng bằng việc lồng nhau và chúng được thay thế bằng những kết quả tính toán để thực hiện chương trình.
- Những lời gọi đệ qui có thể được mô tả dễ dàng.
- Mức độ giống với xử lí song song là cao.
- Tiến trình tính toán hay thủ tục không cần được xét tới.

③ Lập trình logic

Tân từ (thuộc tính)dựa trên sự kiện và suy diễn là cơ sở của lập trình logic. Prolog là một ví dụ về ngôn ngữ lập trình logic.
Đặc trưng

- Bởi vì các sự kiện được mô tả dựa trên logic tân từ, nên tiến trình lập trình này được thực hiện dễ dàng.
- Suy diễn và tính toán logic có thể được thực hiện dễ dàng.

④ Lập trình hướng đối tượng

Trong lập trình hướng đối tượng, hệ thống được xét như một nhóm các đối tượng. Các ngôn ngữ bao gồm Smalltalk, C++, Java và các ngôn ngữ khác.

2.1.2 Thuật toán và cấu trúc dữ liệu

Thuật toán và cấu trúc dữ liệu có quan hệ chặt chẽ với nhau. Ở chừng mực nào đó, cấu trúc dữ liệu xác định ra khuôn khổ cho thuật toán.

(1) Cấu trúc dữ liệu

Cấu trúc dữ liệu được định nghĩa như sau:
Một thủ tục mà chương trình tuân theo để cất giữ dữ liệu và thực hiện những nhiệm vụ đã được giao.

① Cấu trúc dữ liệu cơ sở

Việc lưu giữ dữ liệu nghĩa là lưu giữ dữ liệu vào bộ nhớ chính. Trong lưu giữ dữ liệu vào bộ nhớ chính, kiểu dữ liệu (kiểu dữ liệu, kích cỡ, v.v..) phải được khai báo. Đơn vị cấu trúc dữ liệu cơ sở nhất thường được dùng để khai báo kiểu dữ liệu được gọi là cấu trúc dữ liệu cơ sở. Trong việc thực hiện bước khai báo kiểu dữ liệu này, dữ liệu được thao tác bằng việc dùng tên của dữ liệu có kiểu dữ liệu đã được khai báo trước. (Xem Hình 2-1-1.)

② Cấu trúc dữ liệu hướng vấn đề

Cấu trúc dữ liệu hướng vấn đề được xây dựng bằng việc tổ hợp các cấu trúc dữ liệu cơ sở với một hay nhiều cấu trúc sau:

- Danh sách
- Ngăn xếp
- Hàng đợi
- Cấu trúc cây.

Những phần tử này được xác định bằng việc lập trình (các thuật toán đã được thiết kế). Nếu dữ liệu được xử lí theo thứ tự nó được đưa vào, thì hàng đợi được sử dụng. Nếu dữ liệu được xử lí theo thứ tự đảo với việc đưa vào, thì ngăn xếp được dùng.
Do đó, nội dung của cấu trúc dữ liệu hướng vấn đề được xác định bởi những thuật toán với mức độ nào đó.

Hình 2-1-1 Định nghĩa phần dữ liệu và thủ tục trình chương trình COBOL

Định nghĩa dữ liệu

```
DATA                              DIVISION
FILE                                SECTION
FD   TOUGETU-FILE
01   T-REC
     02  T-HIZUKE       PIC  X(06).
         88  T-ENDF      VALUE  HIGH-VALUE.
     02  FILLER                 PIC  X(34).
FD   RUISEKI-FILE.
01   R-REC.
     02  R-HIZUKE       PIC  X(06).
         88  R-ENDF      VALUE  HIGH-VALUE.
     02  FILLER                 PIC  X(34).
01   N-REC              PIC   X(40).
WORKING-STORAGE   SECTION.
01   T-COUNT            PIC  9(05)
01   R-COUNT            PIC  9(05)
01   N-COUNT            PIC  9(05)
*
```

Phần thủ tục

```
PROCEDURE      DIVISION.
HEIGOU-SYORI.
        OPEN  INPUT  TOUGETU-FILE  RUISEKI-FILE
              OUTPUT  N-RUISEKI-FILE.
        INITIALIZE        T-COUNT  R-COUNT  N-COUNT.
        PERFORM  T-YOMIKOMI.
        PERFORM  R-YOMIKOMI
            PERFORM  UNTIL  T-ENF  AND  R-ENDF.
               IF  T-HIZUKE  <  R-HIZUKE
                 THEN  WRITE  N-REC  FROM  T-REC
                 PERFORM  T-YOMIKOMI
                  ELSE   WRITE N-REC  FROM R-REC
                 PERFORM  R-YOMIKOMI
               END-IF
               COMPUTE  N-COUNT  =  N-COUNT  +  1
            END-PERFORM.
            DISPLAY T-COUNT  R-COUNT  N-COUNT.
            CLOSE  TOUGETU-FILE  RUISEKI-FILE  N-RUISEKI-FILE.
            STOP  RUN.
T-YOMIKOMI.
   READ  TOUGETU-FILE
        AT  END                MOVE  HIGH-VALUE  TO  T-HIZUKE
        NOT AT  END  COMPUTE  T-COUNT  =  T-COUNT  +  1
   END-READ.
R-YOMIKOMI.
   READ  RUISEKI-FILE
     AT  END           MOVE  HIGH-VALUE  TO  R-HIZUKE
     NOT  AT  END   COMPUTE  R-COUNT  =  R-COUNT  +  1
   END-READ
```

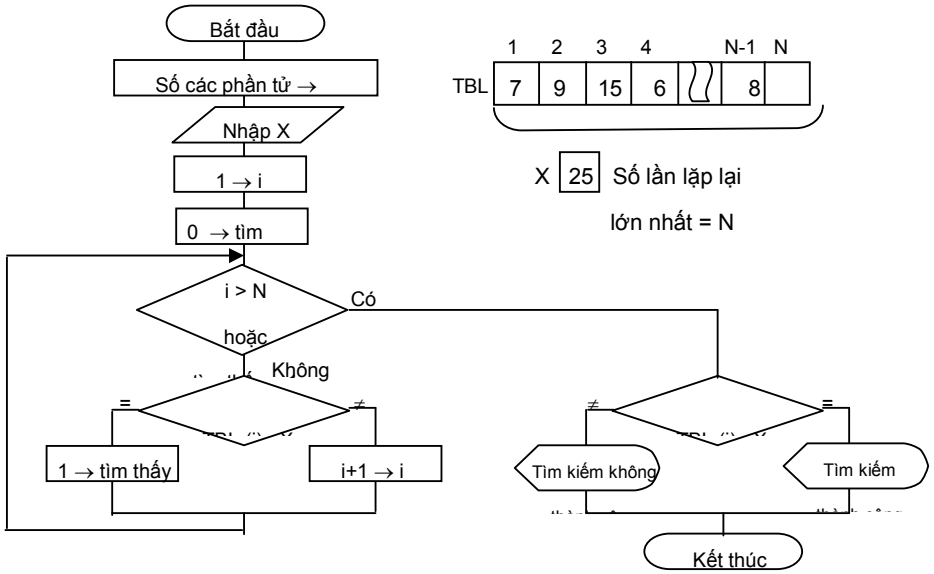
(2) Quan hệ giữa thuật toán và cấu trúc dữ liệu

Mối quan hệ giữa thuật toán và cấu trúc dữ liệu có thể được mô tả như sau:

① Xử lí mảng

Lấy thuật toán duyệt tuyến tính (chi tiết được giải thích trong Mục 2.2.1) được vẽ trong Hình 2-1-2 làm ví dụ.

Hình 2-1-2 Thuật toán duyệt tuyến tính và cấu trúc dữ liệu



Duyệt tuyến tính thường được sử dụng nhiều nhất để duyệt dữ liệu. Trong khi thực hiện duyệt tuyến tính trên dữ liệu, dữ liệu được duyệt trong khi chỉ số trong mảng được tăng lên. Số tối đa lần duyệt được lặp lại là kích cỡ của mảng. Tức là, nếu cấu trúc dữ liệu mảng được dùng, thì thủ tục và số lần lặp lại là được xác định.

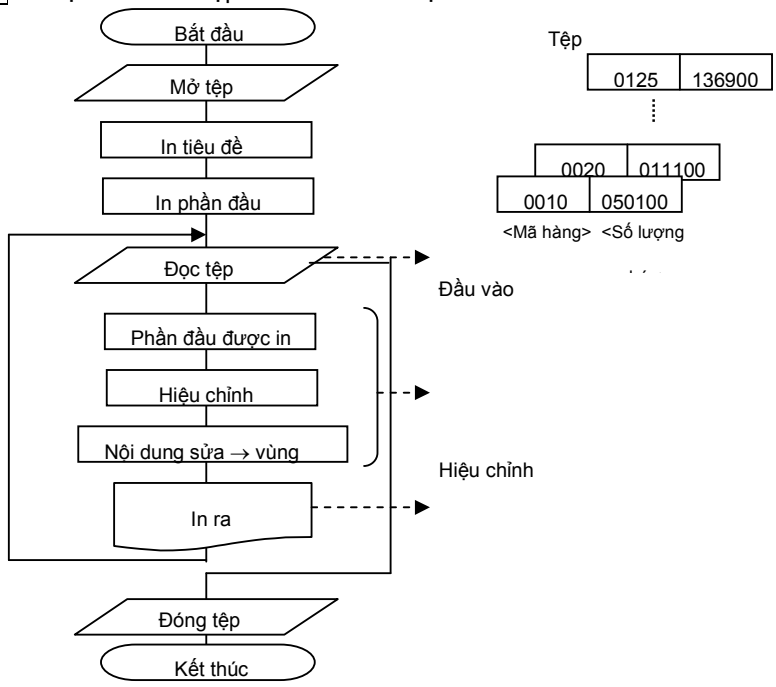
② Xử lí tệp

Lấy thuật toán để đọc và in tệp như được nêu trong Hình 2-1-3 làm ví dụ. (Chi tiết được giải thích trong mục 2.2.5.)

Việc xử lí đọc, soạn thảo và in một tệp được lặp lại cho tới khi không còn dữ liệu trong tệp.

Vì tệp phải được truy nhập trong từng chu trình xử lí nên nhiệm vụ này được thực hiện dưới dạng một chu trình.

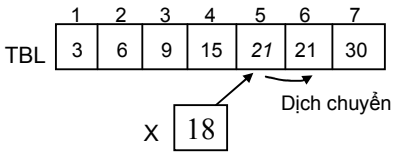
Hình 2-1-3 Thuật toán xử lí tệp và cấu trúc dữ liệu



③ Xử lí danh sách

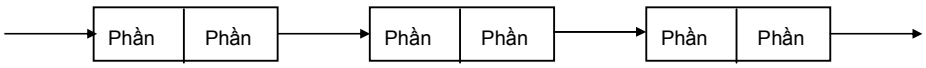
Trong xử lí cấu trúc mảng, dữ liệu có thể được duyệt và cập nhật một cách trơn tru nhưng lại mất thời gian để chèn thêm hay xoá dữ liệu. Bởi vì dữ liệu được thu xếp trong hàng đợi, nên việc chèn thêm hay xoá dữ liệu không tránh khỏi đi kèm với việc dịch chuyển dữ liệu ra sau hay lên trước.

Hình 2-1-4 Chèn thêm dữ liệu vào trong mảng



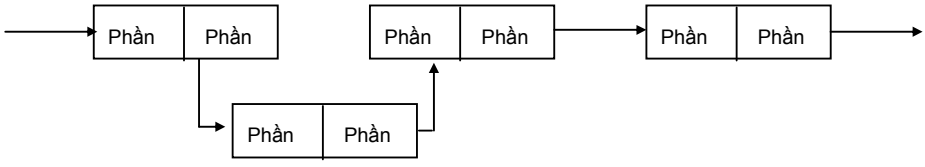
Dùng cấu trúc danh sách, việc chèn thêm hay xoá dữ liệu là dễ dàng. Mặc dầu dữ liệu được thu xếp theo cách có trật tự trong cấu trúc danh sách, nó được thu xếp một cách logic để cho không cần phải được thu xếp về mặt vật lí theo thứ tự tuần tự.

Hình 2-1-5 Cấu trúc danh sách



Trong trường hợp của cấu trúc danh sách, dữ liệu không nhất thiết phải bị dịch chuyển như được vẽ trong Hình 2-1-6; dữ liệu có thể được chèn thêm hay xoá đi bởi việc thao tác con trỏ một cách đơn giản.

Hình 2-1-6 Chèn thêm dữ liệu vào trong danh sách



2.2 Các thuật toán

Thuật toán nên được xem như để giải quyết từng bài toán đặc biệt. Nếu thuật toán có thể giải quyết các bài toán tương tự đã được thiết kế và đã có sẵn, thì việc dùng thuật toán như vậy sẽ tạo khả năng cho bạn tạo ra thuật toán tốt hơn theo cách hiệu quả.
Mục này mô tả về các thuật toán tiêu biểu đã được phát triển cho tới nay trong quan hệ với từng kiểu bài toán.

2.2.1 Thuật toán duyệt

Việc duyệt bảng là phương pháp thực hiện việc duyệt trên bảng và tệp được lưu giữ trong bộ nhớ để tìm ra các yếu tố đáp ứng cho các yêu cầu xác định.
Mục này mô tả cho hai phương pháp duyệt bảng: phương pháp duyệt tuyến tính (hay tuần tự) và phương pháp duyệt nhị phân.

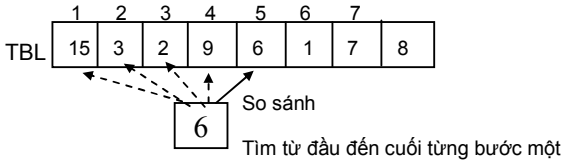
(1) Phương pháp duyệt tuyến tính (hay tuần tự)

Phương pháp duyệt tuyến tính (hay tuần tự) là phương pháp duyệt rất đơn giản từ đầu bảng theo cách tuần tự.

① Phương pháp duyệt vét cạn

Phương pháp duyệt vét cạn là phương pháp duyệt đơn giản nhất cho một bảng đi từ đầu đến cuối.

Hình 2-2-1 Hình ảnh về phương pháp duyệt vét cạn

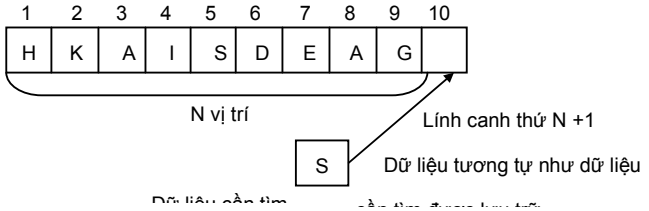


Dữ liệu tìm kiếm sẽ được đối chiếu với dữ liệu trong bảng. Thuật toán này được thiết kế sao cho việc duyệt là thành công chỉ nếu có dữ liệu sánh đúng với dữ liệu cần tìm và nó là không thành công nếu không có dữ liệu nào sánh đúng với dữ liệu cần tìm.
Thủ tục duyệt chấm dứt khi việc sánh thành công đầu tiên xuất hiện. Do đó, phương pháp duyệt này là không thích hợp cho việc đếm số dữ liệu sánh đúng với dữ liệu cần tìm.

② Phương pháp duyệt lính canh

Phương pháp duyệt lính canh dùng thuật toán mà trong đó cùng dữ liệu (lính canh) như dữ liệu cần tìm được đặt vào cuối của bảng để làm đơn giản hoá thuật toán duyệt và để làm giảm số lần dữ liệu được so sánh.

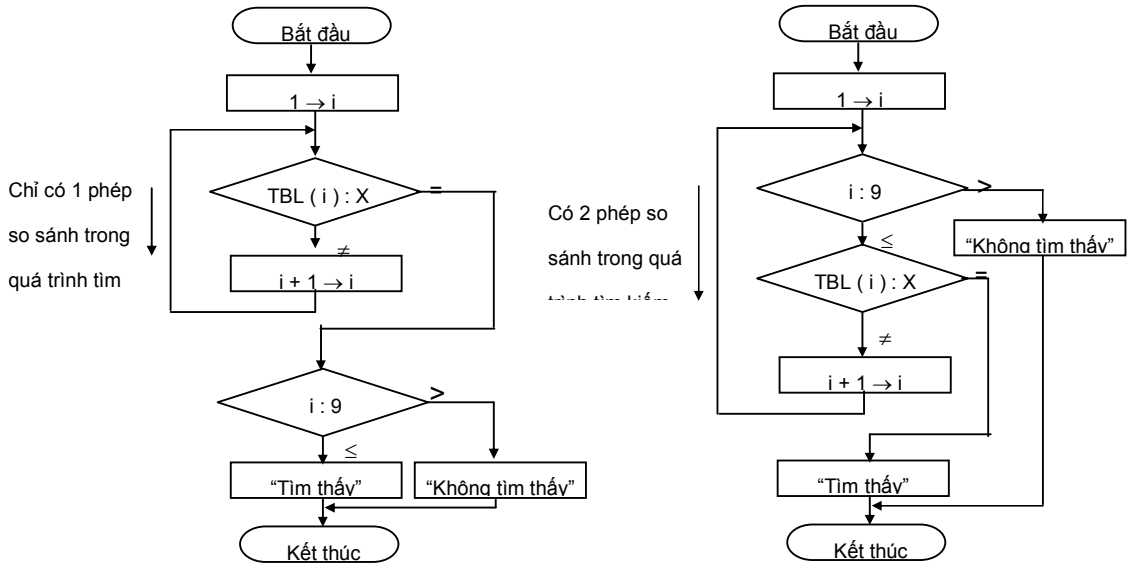
Hình 2-2-2 Phương pháp duyệt lính canh



Nếu số dữ liệu được chứa trong bảng là N, thì cũng dữ liệu đó (lính canh) như dữ liệu cần tìm được lưu giữ vào vị trí (N + 1) sao cho dữ liệu cần tìm có thể được đối sánh ngay lập tức với dữ liệu lính canh.

Hình 2-2-3 chỉ ra việc so sánh của trường hợp phương pháp duyệt lính canh được dùng và trường hợp nó không được dùng.

Hình 2-2-3 So sánh trường hợp phương pháp duyệt lính canh được dùng và trường hợp nó không được dùng



<Trường hợp phương pháp duyệt lính canh được dùng>

Trong vòng thứ nhất của việc đối chiếu dữ liệu, dữ liệu cần tìm được xác thực. Trong vòng thứ hai, phải xác định liệu có dữ liệu sánh đúng với dữ liệu cần tìm hay không. Tức là, nếu số dữ liệu là N, việc so sánh được thực hiện chỉ (N + 1) lần.

<Trường hợp phương pháp duyệt lính canh không được dùng>

Trong một vòng đối chiếu dữ liệu, tính xác thực của dữ liệu cần tìm cũng như liệu việc tìm kiếm có kết thúc hay không phải được xác định. Tức là, việc so sánh được thực hiện (N × 2) lần.

<Số lần so sánh cực đại được thực hiện nếu số phần tử là N:>

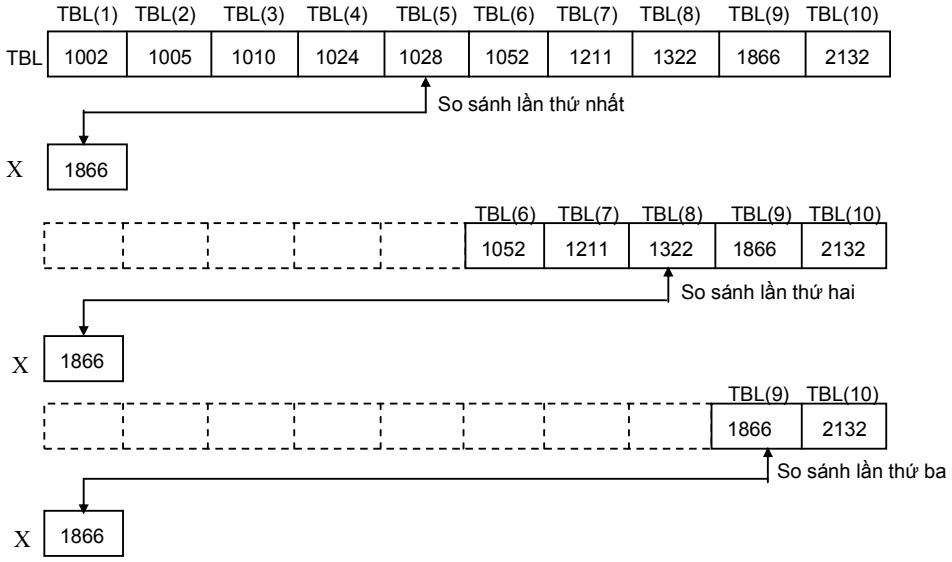
- (N + 1) lần nếu phương pháp duyệt lính canh được dùng
- (N × 2) lần nếu phương pháp duyệt vét cạn được dùng

(2) Phương pháp duyệt nhị phân

Phương pháp duyệt nhị phân là phương pháp làm hẹp dần dữ liệu đích khi phân chia liên tiếp miền duyệt thành hai phần. Số các phép so sánh có thể được giảm đi rất nhiều nếu so với phương pháp duyệt tuyến tính và tính hiệu quả duyệt có thể được nâng cao. Tuy nhiên phương pháp duyệt này đòi hỏi rằng các phần tử phải được sắp theo thứ tự tăng hay giảm.
Hình 2-2-4 chỉ ra thuật toán được dùng cho phương pháp duyệt nhị phân.

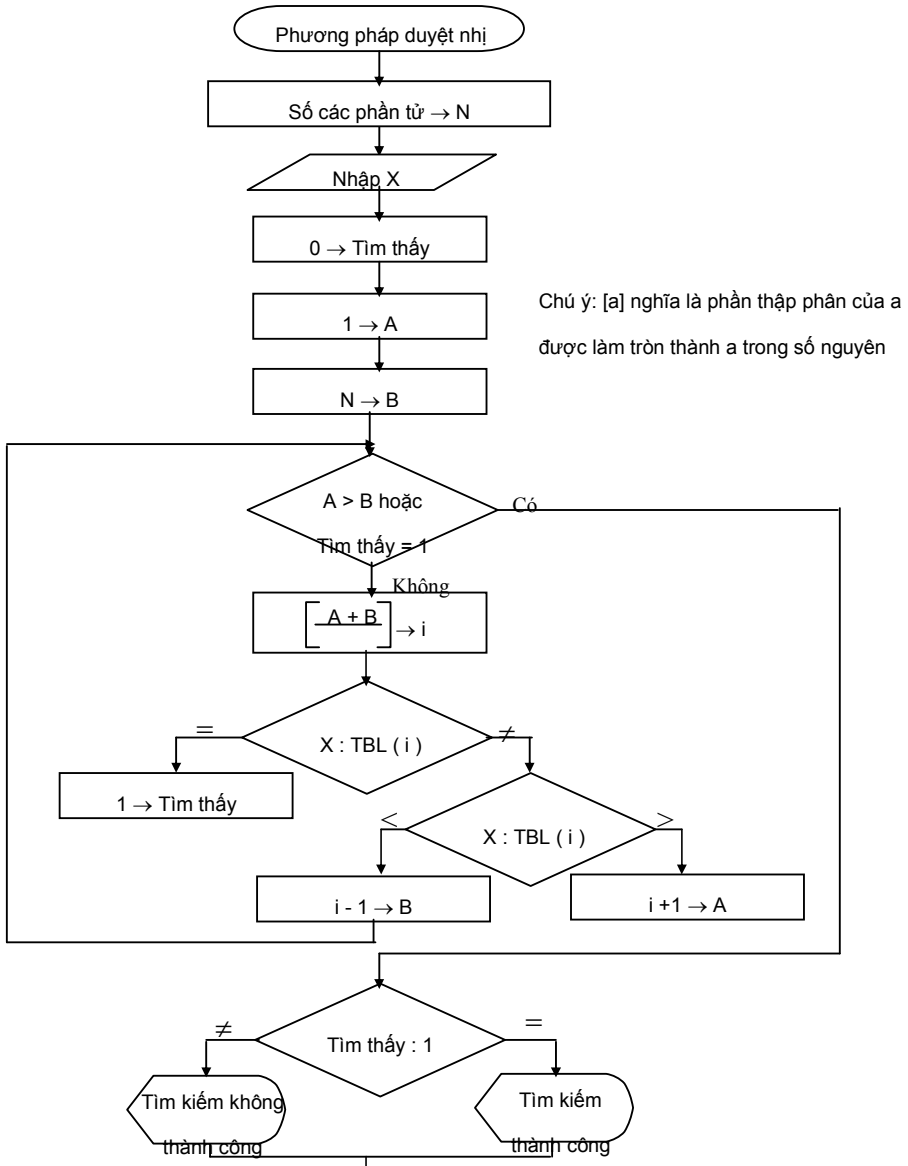
Hình 2-2-4 Thuật toán cho phương pháp duyệt nhị phân

Bước 1: Lấy tổng chỉ số dưới ở đầu bảng và chỉ số dưới ở cuối bảng chia 2.
Bước 2: Phần tử có giá trị ở bước 1 được so sánh với phần tử đích.
Bước 3: Nếu có phần tử tương tự với phần tử đích thì kết thúc tìm kiếm.
Bước 4: Nếu giá trị của phần tử đích nhỏ hơn giá trị của phần tử trong bảng, chỉ số được trừ đi 1 và giá trị được sử dụng như chỉ số biểu diễn cuối bảng.
Nếu giá trị của phần tử gốc lớn hơn giá trị của phần tử trong bảng thì chỉ số hiện tại được cộng thêm 1 và lấy làm giá trị chỉ số của đầu bảng.



Vì các phần tử được xếp theo thứ tự tăng hay giảm, nên dữ liệu nhỏ hơn dữ liệu tham chiếu không cần phải được duyệt nếu dữ liệu đang được duyệt lớn hơn dữ liệu tham chiếu. Do đó, số dữ liệu cần duyệt có thể được giảm đi một nửa sau lần duyệt thứ nhất - một ưu thế lớn về hiệu quả duyệt.
Hình 2-2-5 chỉ ra sơ đồ khối của phương pháp duyệt nhị phân.

Hình 2-2-5 Lưu đồ của phương pháp duyệt nhị phân



<Số lần tối đa và số lần trung bình được thực hiện: Nếu số phân tử là N>

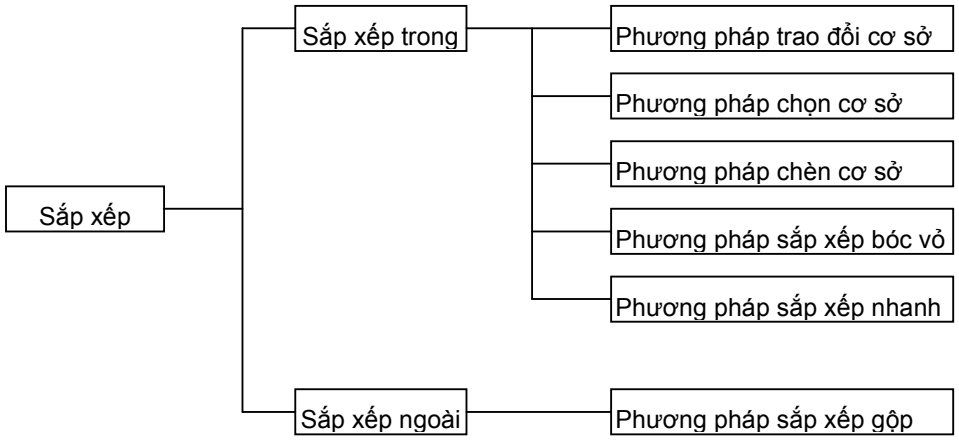
- Số lần trung bình = $[\log_2 N]$
- Số lần tối đa $-[\log_2 N] + 1$

([ ] là kí hiệu Gaussian và phần thập phân của giá trị trong kí hiệu này được làm tròn)

2.2.2 Thuật toán sắp xếp

Việc sắp xếp dữ liệu là việc tổ chức lại dữ liệu theo một thứ tự đặc biệt. Việc sắp xếp theo thứ tự giá trị từ nhỏ tới lớn được gọi là sắp xếp theo thứ tự tăng, còn việc sắp dữ liệu theo chiều ngược lại được gọi là sắp theo thứ tự giảm.
Nếu dữ liệu được sắp xếp hay tổ chức theo một thứ tự đặc biệt nào đó (tiền lương, mã hàng hoá v.v..), thì hiệu quả của công việc xử lí dữ liệu có thể được tăng lên. Do đó, thuật toán sắp xếp là một trong những thuật toán thông dụng nhất. Dùng thuật toán sắp xếp này, người ta có thể sắp xếp số và kí tự. Kiểu sắp xếp này là có thể bởi vì các kí tự được biểu diễn như các mã kí tự (mã nội bộ) trong máy tính.
Hình 2-2-6 đưa ra các phương pháp sắp xếp khác nhau.

Hình 2-2-6 Phương pháp sắp xếp



Sắp xếp trong nghĩa là sắp xếp dữ liệu trong bộ nhớ chính. Sắp xếp ngoài nghĩa là sắp xếp dữ liệu đã được lưu giữ trên đĩa từ và đơn vị nhớ phụ khác.
Trong việc chọn một phương pháp sắp xếp, tốc độ sắp xếp, cũng như cách dữ liệu được sắpxếp phải được xem xét để làm tối thiểu thời gian cần cho việc so sánh và tráo đổi dữ liệu. Trong việc chọn một phương pháp, bạn cũng phải làm rõ thời gian tính toán (độ phức tạp tính toán). Điểm này sẽ được mô tả chi tiết trong Mục 2.3.

(1) Phương pháp tráo đổi cơ sở (sắp xếp nổi bọt - bubble sort)

Phương pháp tráo đổi cơ sở (sắp xếp nổi bọt) được dùng để so sánh một cặp dữ liệu tuần tự từ đầu mảng. Nếu trật tự mà sai, thì dữ liệu được tráo đổi. Khi tất cả các khoản mục dữ liệu trong một mảng được so sánh, thì trình này cho lại đầu mảng và nó được lặp lại cho tới khi không còn khoản mục dữ liệu nào cần tráo đổi nữa. Phương pháp này là phương pháp sắp xếp đơn giản nhất, nổi tiếng nhất. Cái tên "sắp xếp nổi bọt" được cho bởi vì việc chuyển tối đa hay tối thiểu dữ liệu giống như bọt nổi lên bề mặt nước.

Hình 2-2-7 Các bước của phương pháp tráo đổi cơ sở

Thuật toán sắp xếp dữ liệu theo thứ tự tăng dần

Bước 1: So sánh phần tử thứ nhất và thứ hai trong bảng.

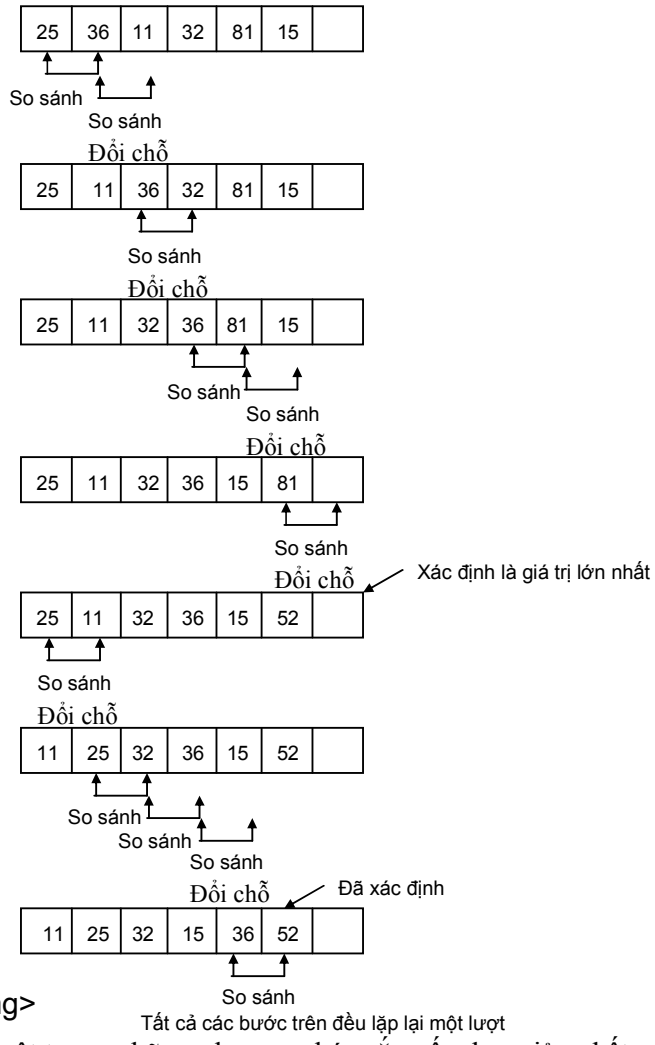
Bước 2: Nếu phần tử thứ nhất lớn hơn phần tử thứ hai thì đổi chỗ phần tử thứ nhất và phần tử thứ hai.

Bước 3: Nếu phần từ thứ nhất nhỏ hơn phần tử thứ hai thì không đổi chỗ hai phần tử.

Bước 4: So sánh phần tử thứ hai và phần tử thứ ba và lặp lại bước 2 và bước 3.

Bước 5: Lặp lại cho tới khi tới phần tử cuối cùng trong bảng. Khi tìm tới phần tử cuối cùng, giá trị lớn nhất được lưu trữ trong phần tử cuối cùng trong bảng.

Bước 6: Thực hiện bước 1 tới bước 4 và bước 5 cho tới khi phép toán đi tới phần tử cuối cùng



<Đặc trưng>

- Đây là một trong những phương pháp sắp xếp đơn giản nhất
- Tính hiệu quả là thấp vì dữ liệu được so sánh vô điều kiện ngay cả khi chúng đã được sắp xếp đúng.
- Nếu khối lượng dữ liệu lớn thì tốn thời gian xử lí.

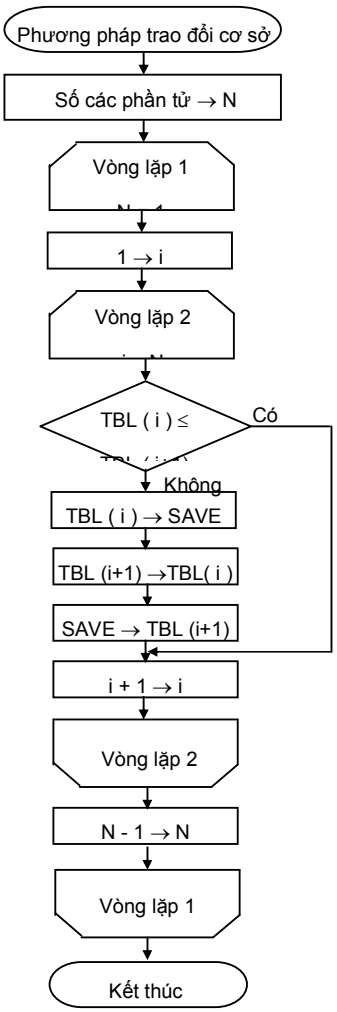
<Độ phức tạp tính toán>

- Độ phức tạp tính toán tối đa: $O(n^2)$

- Độ phức tạp tính toán trung bình: $O(n^2)$

Hình 2-2-8 đưa ra lưu đồ của phương pháp tráo đổi cơ sở.

Hình 2-2-8 Lưu đồ của phương pháp tráo đổi cơ sở



(2) Phương pháp lựa cơ sở (basic selection sort)

Trong thuật toán sắp xếp của phương pháp lựa cơ sở, một khoản mục dữ liệu với giá trị nhỏ nhất (hay lớn nhất) được chọn ra đầu tiên từ tất cả các khoản mục dữ liệu và nó được tráo đổi với khoản mục dữ liệu ở đầu mảng, rồi cùng việc này được thực hiện lặp lại trên tất cả các khoản mục dữ liệu còn lại. Khi dữ liệu đúng đưa vào vị trí cuối cùng là một, thì việc sắp xếp dữ liệu được hoàn thành.
Vì phương pháp này cho phép phần tử còn lại với giá trị nhỏ nhất hay lớn nhất được chọn lựa, nên nó được gọi là phương pháp lựa cơ sở.
Hình 2-2-9 đưa ra thuật toán của phương pháp lựa cơ sở.

Hình 2-2-9 Các bước của phương pháp lựa cơ sở

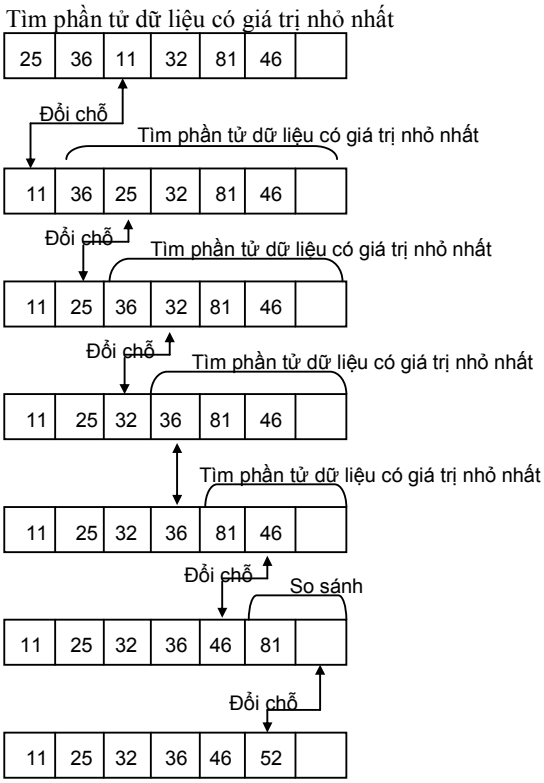
Các bước trong thuật toán sắp xếp dữ liệu theo thứ tự tăng dần

Bước 1: Phát hiện phần tử dữ liệu có giá trị nhỏ nhất trong bảng.

Bước 2: Trao đổi phần tử dữ liệu có giá trị nhỏ nhất với phần tử dữ liệu đầu tiên trong bảng.

Bước 3: Phát hiện phần tử dữ liệu có giá trị nhỏ nhất trong các phần tử còn lại trong bảng.

Bước 4: Trao đổi phần tử dữ liệu có giá trị nhỏ nhất trong bước 3 với phần



<Đặc trưng>

- Một trong các phương pháp sắp xếp đơn giản nhất, như phương pháp tráo đổi
- Tính hiệu quả thấp vì khoản mục dữ liệu được so sánh vô điều kiện ngay cả khi chúng

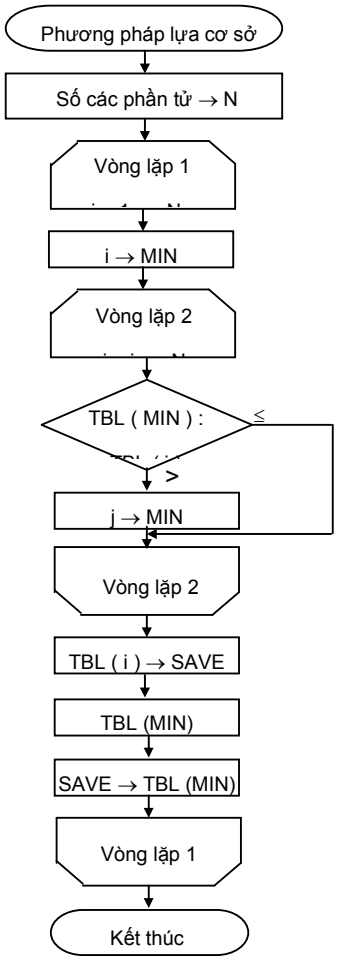
được sắp xếp đúng.
- Nếu khối lượng dữ liệu lớn, thì tốn thời gian xử lí.

<Độ phức tạp tính toán>

- Độ phức tạp tính toán tối đa: $O(n^2)$
- Độ phức tạp tính toán trung bình: $O(n^2)$

Hình 2-2-10 đưa ra lưu đồ của phương pháp lựa cơ sở.

Hình 2-2-10 Lưu đồ của phương pháp lựa cơ sở



(3) Phương pháp chèn cơ sở (basic insert sort)

Trong thuật toán của phương pháp chèn cơ sở, trong khi các khoản mục dữ liệu được sắp, một khoản mục dữ liệu chưa sắp xếp được chèn vào vị trí đúng trong trình tự các khoản mục dữ liệu đã được sắp xếp.
Hình 2-2-11 đưa ra thuật toán cho phương pháp chèn cơ sở.

Hình 2-2-11 Các bước của phương pháp chèn cơ sở

Thuật toán sắp xếp dữ liệu theo thứ tự tăng dần

Bước 1: So sánh phần tử thứ nhất và thứ 2 trong bảng.

Bước 2: Nếu phần tử thứ nhất nhỏ hơn phần tử thứ hai, giữ nguyên trật tự.

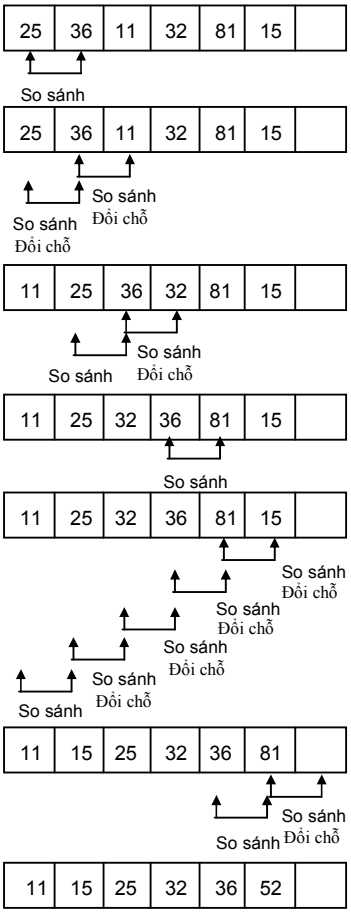
Bước 3: Nếu phần tử thứ hai nhỏ hơn phần tử thứ nhất, đổi chỗ phần tử thứ hai và thứ nhất.

Lúc này phần tử thứ nhất và phần tử thứ hai đã theo đúng trật tự.

Bước 4:So sánh phần tử thứ hai và phần tử thứ ba.

Bước 5: Nếu phần tử thứ hai nhỏ hơn phần tử thứ ba, giữ nguyên trật tự.

Bước 6: Nếu phần tử thứ ba nhỏ hơn phần tử thứ hai, đổi chỗ phần tử thứ hai và phần tử thứ ba.

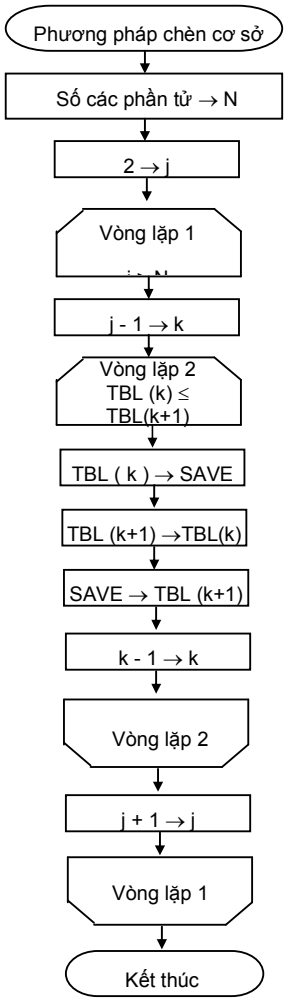


<Đặc trưng>

- Một trong các phương pháp sắp xếp đơn giản nhất, như phương pháp tráo đổi
- Vì dữ liệu đứng trước đã được sắp nên tốc độ so sánh và chèn thêm là nhanh.
- Nếu khối lượng dữ liệu lớn, tốn thời gian xử lí.

Hình 2-2-12 đưa ra lưu đồ của phương pháp chèn cơ sở.

Hình 2-2-12 Lưu đồ của phương pháp chèn cơ sở



(4) Phương pháp sắp xếp sàng lắc (shaker sort)

Thuật toán của phương pháp sắp xếp sàng lắc về cơ bản là giống như phương pháp tráo đổi

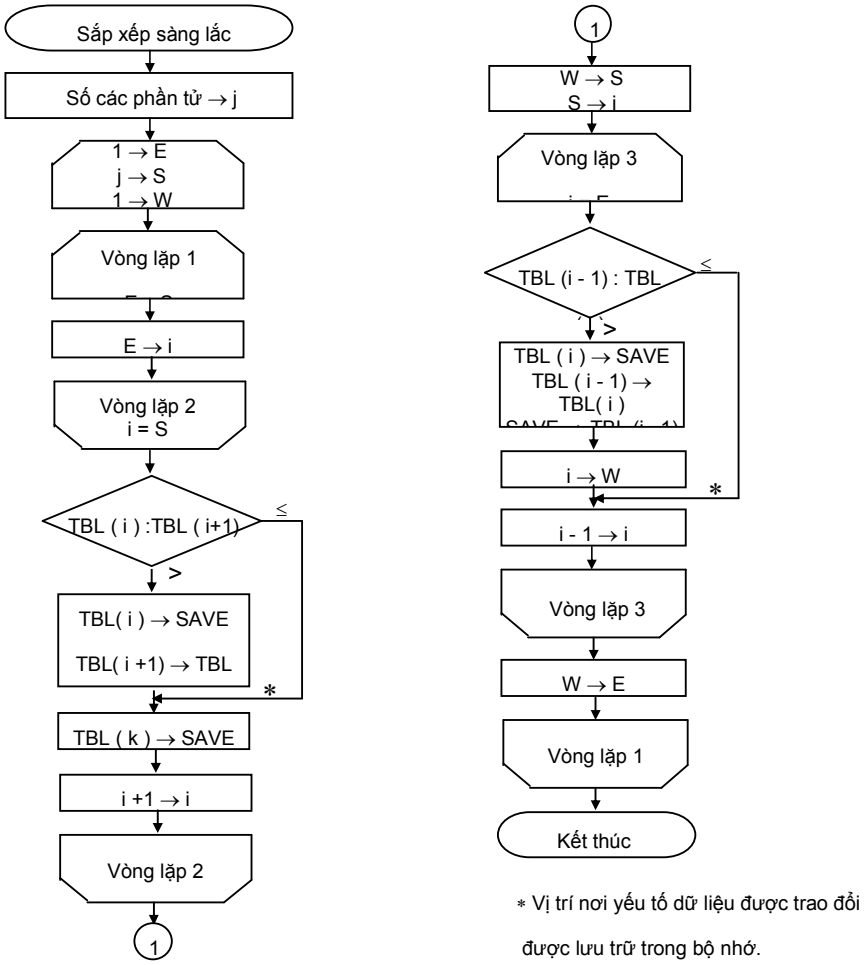
cơ sở (sắp xếp nổi bọt). Trong thuật toán của phương pháp tráo đổi cơ sở, các khoản mục dữ liệu được so sánh từ trái sang phải và được sắp theo cách giá trị cực đại (cực tiểu) được đặt vào vị trí bên phải nhất. Tuy nhiên, trong thuật toán của phương pháp sắp xếp sàng lắc, các khoản mục dữ liệu trước hết được so sánh từ trái sang phải, sau đó, sau khi giá trị cực đại (cực tiểu) đã được đặt vào vị trí bên phải nhất, thì các khoản mục được so sánh từ phải sang trái và giá trị cực tiểu (cực đại) được đặt vào vị trí bên trái nhất; thao tác này được thực hiện lặp lại để sắp xếp dữ liệu.

<Đặc trưng>

- Nếu khối lượng dữ liệu lớn, thì tốn thời gian.

Hình 2-2-13 đưa ra lưu đồ của phương pháp sắp xếp sàng lắc.

Hình 2-2-13 Lưu đồ của phương pháp sắp xếp sàng lắc

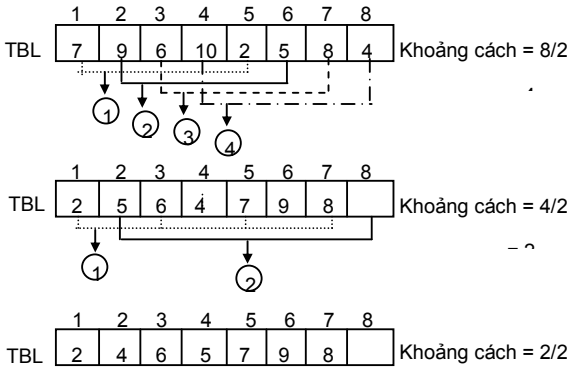


* Vị trí nơi yếu tố dữ liệu được trao đổi được lưu trữ trong bộ nhớ.

(5) Phương pháp sắp xếp bóc vỏ (Shell sort method)

Phương pháp sắp xếp bóc vỏ là phiên bản mở rộng của phương pháp chèn cơ sở. Hai khoản mục dữ liệu ở xa nhau vào một khoảng nào đó được lấy ra để so sánh với nhau. Khoảng này gọi là lỗ hổng. Lỗ hổng này được đặt lớn lúc bắt đầu sắp xếp; khi việc sắp xếp tiến triển, nó dần được làm nhỏ lại và cuối cùng được đặt là 1. Có những cách khác để xác định lỗ hổng này. Nếu số dữ liệu là n, một cách đơn giản để xác định lỗ hổng là [n/2], [n/4],⋯ 1. Khi lỗ hổng này cuối cùng được đặt là 1, thì việc sắp xếp được thực hiện theo đích xác cùng cách như phương pháp chèn cơ sở.
Dùng phương pháp chèn cơ sở, các phần tử kề nhau được so sánh và tráo đổi và, do đó, tốc độ thực hiện thấp. Dùng phương pháp sắp xếp bóc vỏ, các mẩu dữ liệu ở xa nhau và nằm ở những vị trí khác nhau nhanh chóng được tráo đổi để cho các khoản mục dữ liệu được sắp sai vị trí có thể được sắp lại về vị trí đúng trong những giai đoạn sớm nhất của thao tác sắp xếp. Khi việc sắp xếp tiến hành, lỗ hổng giữa các khoản mục dữ liệu cần được so sánh sẽ hẹp dần.
Hình 2-2-14 đưa ra thuật toán của phương pháp sắp xếp bóc vỏ.

Hình 2-2-14 Các bước của phương pháp sắp xếp bóc vỏ

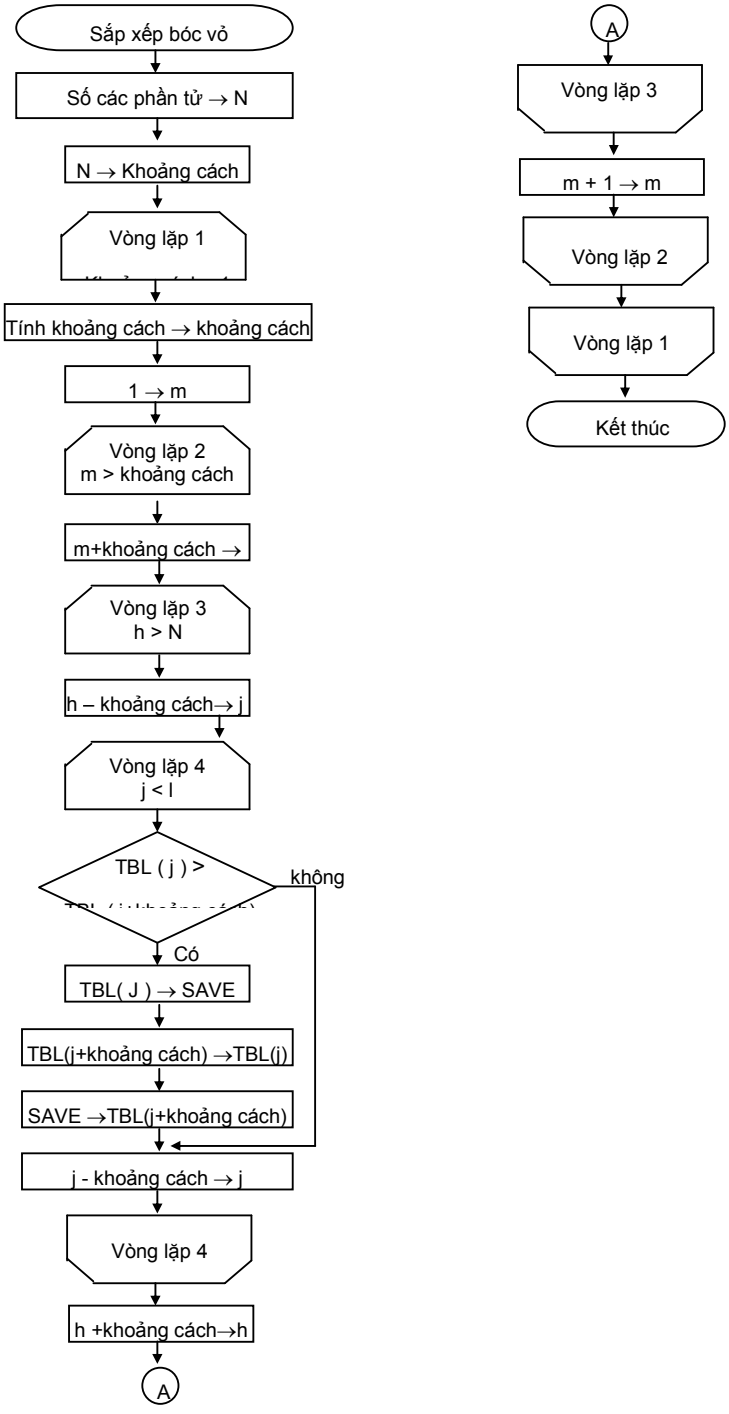


<Đặc trưng>

- Nếu một phần của dữ liệu đã được sắp, thì việc sắp xếp có thể được hoàn tất rất nhanh chóng.
- Phương pháp sắp xếp tốc độ cao dùng phương pháp chèn cơ sở

Hình 2-2-15 đưa ra lưu đồ của phương pháp sắp xếp bóc vỏ.

Hình 2-2-15 Lưu đồ của phương pháp sắp xếp bóc vỏ



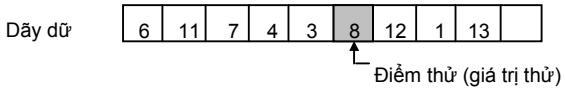
(6) Phương pháp sắp xếp nhanh (Quick sort method)

Phương pháp sắp xếp nhanh do Hoare thiết kế ra. Nó hiện thời là phương pháp sắp xếp nhanh nhất dùng phương pháp gọi đệ qui.

<Sắp xếp dữ liệu theo thứ tự tăng>

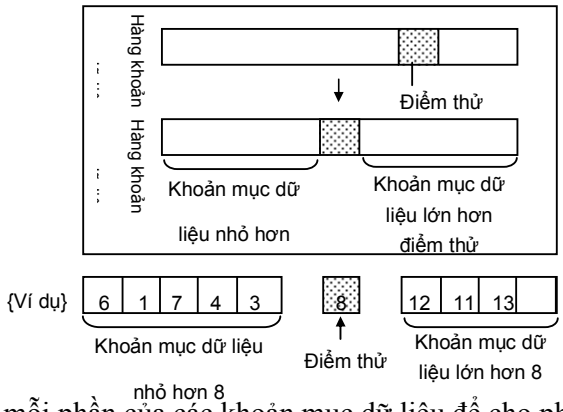
1. Một giá trị tham chiếu (điểm thử hay giá trị thử) được chọn từ dữ liệu cần sắp xếp. Mặc dầu các giá trị khác nhau được dùng như giá trị tham chiếu, một giá trị trung gian của ba phần tử (phần tử phải, giữa và trái) thường được dùng. Chúng ta dùng giá trị trung gian trong ví dụ sắp xếp được nêu dưới đây.

Hình 2-2-16 Thử (giá trị thử)



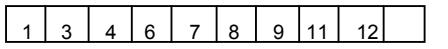
2. Những khoản mục dữ liệu nhỏ hơn giá trị thử được chuyển sang bên trái của khoản mục dữ liệu thử trong khi các khoản mục dữ liệu lớn hơn giá trị thử được chuyển sang bên phải của nó. Tất cả các khoản mục dữ liệu vậy được chia thành hai phần.

Hình 2-2-17 Chuyển dữ liệu



3. Điểm thử được chọn từ mỗi phần của các khoản mục dữ liệu để cho phần này được phân chia thêm nữa thành hai phần con.
4. Thao tác phân chia này được lặp lại cho tới khi chỉ còn lại một phần tử.

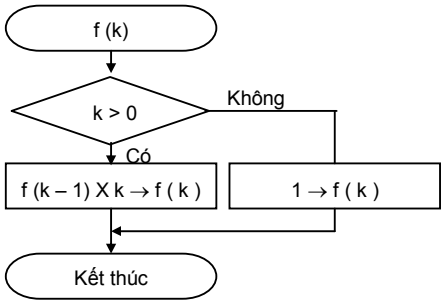
Hình 2-2-18 Dữ liệu sau khi việc sắp xếp được hoàn tất



Phương pháp phân chia một vấn đề lớn thành những vấn đề nhỏ và giải quyết từng vấn đề nhỏ một cách riêng rẽ, như phương pháp sắp xếp nhanh này, được gọi là phương pháp chia và trị.

Trong khi thực hiện các bước 3 và 4 trên, phương pháp gọi đệ qui tự gọi tới chính nó nói chung được sử dụng. Phương pháp gọi đệ qui này thường được dùng để tính giai thừa. (Xem Hình 2-2-19.)

Hình 2-2-19 Thuật toán tính giai thừa



Về độ phức tạp tính toán, nếu trường hợp lí tưởng, điểm thử có thể chia dữ liệu thành hai phần bao giờ cũng có thể chọn được, thì số lần so sánh sẽ rất gần với O ($n$log$n$). Nếu giá trị cực đại (cực tiểu) trong dãy dữ liệu bao giờ cũng được chọn làm điểm thử, thì số lần so sánh sẽ là tồi nhất, O (n^2). Mặc dầu độ phức tạp tính toán thường chỉ ra độ phức tạp tính toán cực đại, trường hợp như thế này có thể xảy ra mà độ phức tạp tính toán trung bình trở thành một chỉ báo quan trọng.

Sau khi các khoản mục dữ liệu được phân chia thành hai phần, các khoản mục dữ liệu trong mỗi phần là chủ đề cho một trình đệ qui có thể được xử lí riêng biệt. Do đó, việc xử lí song song có thể được thực hiện. Thời gian xử lí trung bình là O (log$n$) nếu việc xử lí song song được thực hiện.

Hình 2-2-20 đưa ra thuật toán để thực hiện việc sắp xếp nhanh chóng với tập thử ở vị trí bên phải nhất của mảng.

Đặc biệt, hai con trỏ được dùng và trình này tiến hành từ cả hai đầu đi vào trung tâm của dãy dữ liệu. Một con trỏ đi từ trái sang phải là "i" và một con trỏ đi từ phải sang trái là "j."

Hình 2-2-20 Thuật toán về phương pháp sắp xếp nhanh

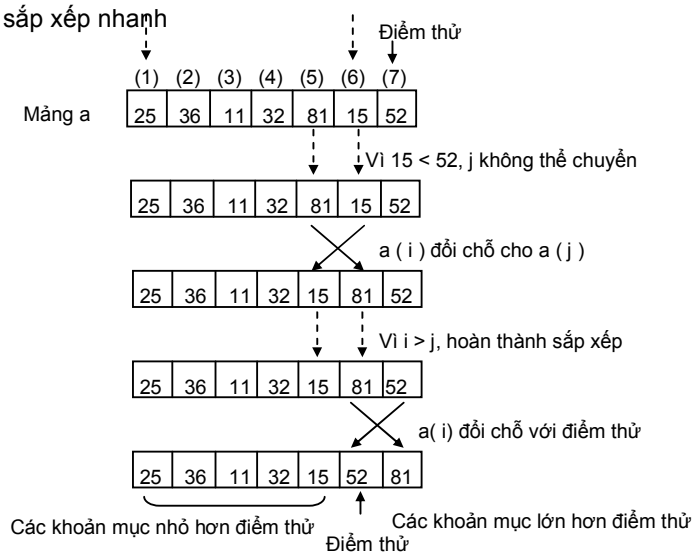
Bước 1: Đây là bước chuẩn bị, khởi chạy các con trỏ i và j.
- i tại vị trí cuối cùng bên trái và bằng 1
- j tại vị trí cuối cùng bên phải nhưng a(7) của điểm chuẩn là 6

Bước 2-1: i được chuyển sang phải cho tới khi tìm ra khoản mục lớn hơn điểm chuẩn.

Bước 2-2: j được chuyển sang trái cho tới khi tìm ra khoản mục nhỏ hơn điểm chuẩn.

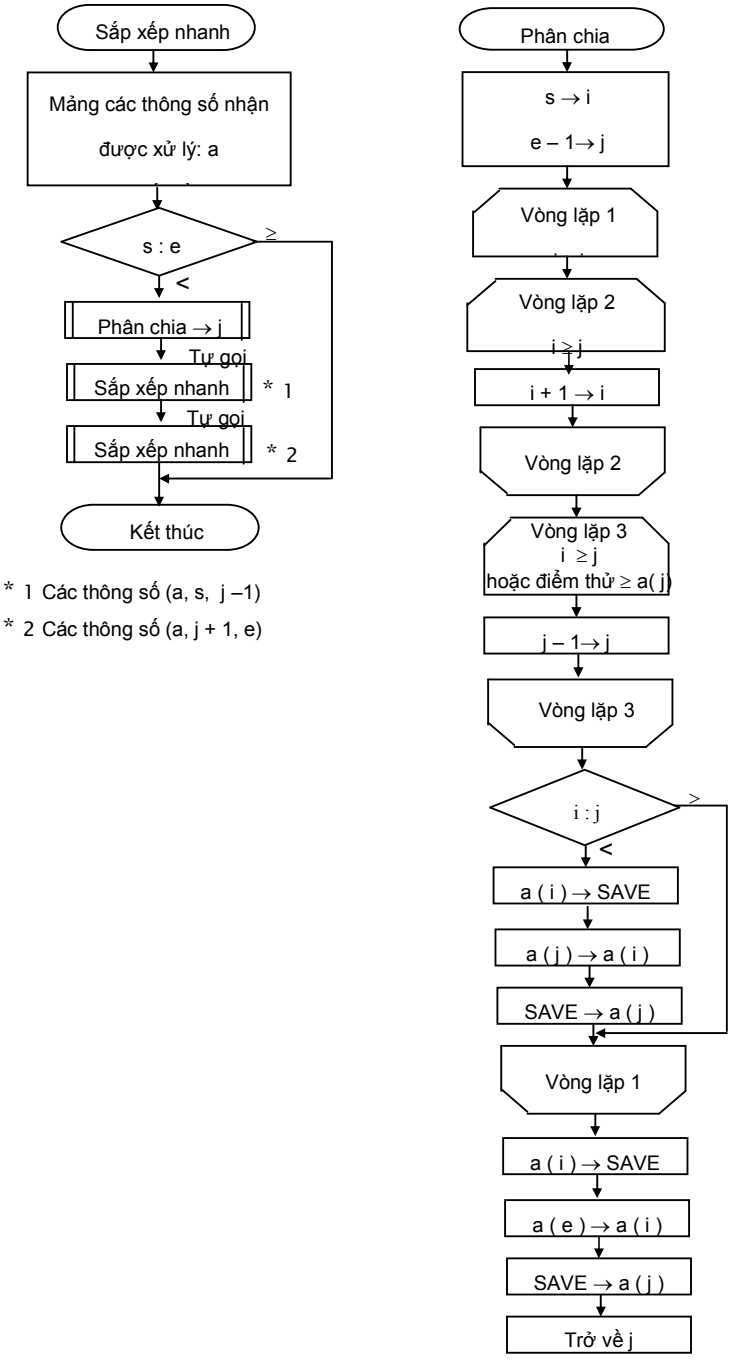
Bước 2-3: đổi chỗ a (i) và a (j)

Bước 2-4: Lặp lại bước 2-1, 2-2, 2-3 cho tới khi



Hình 2-2-21 đưa ra lưu đồ của phương pháp sắp xếp nhanh.

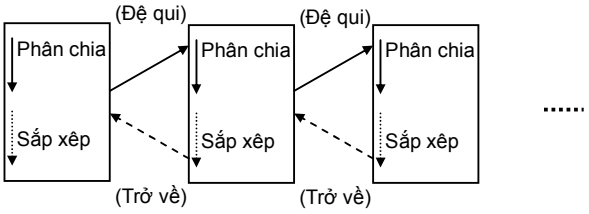
Hình 2-2-21 Thuật toán của phương pháp sắp xếp nhanh



(7) Phương pháp sắp xếp gộp (Merge sort method)

Trong thuật toán của phương pháp sắp xếp gộp, tất cả các khoản mục dữ liệu được phân chia thành các phần và việc sắp xếp được thực hiện theo từng phần đã được chia ra. Các phần được sắp xếp được gộp vào trong dãy được sắp xếp (xem Hình 2-2-22). Gộp nghĩa là so sánh các các khoản mục dữ liệu trong hai dãy được sắp xếp từ đầu và tạo ra một dãy dữ liệu được sắp xếp bằng việc lấy ra lặp đi lặp lại khoản mục dữ liệu nhỏ hơn theo cách tuần tự. Nếu số khoản mục dữ liệu là $2n$, lặp việc gộp n lần sẽ hoàn thành việc sắp xếp. Nếu nó không phải là $2n$, thì cần sự điều chỉnh nào đó.

Hình 2-2-22 Biểu đồ khái niệm về sắp xếp gộp



<Đặc trưng>

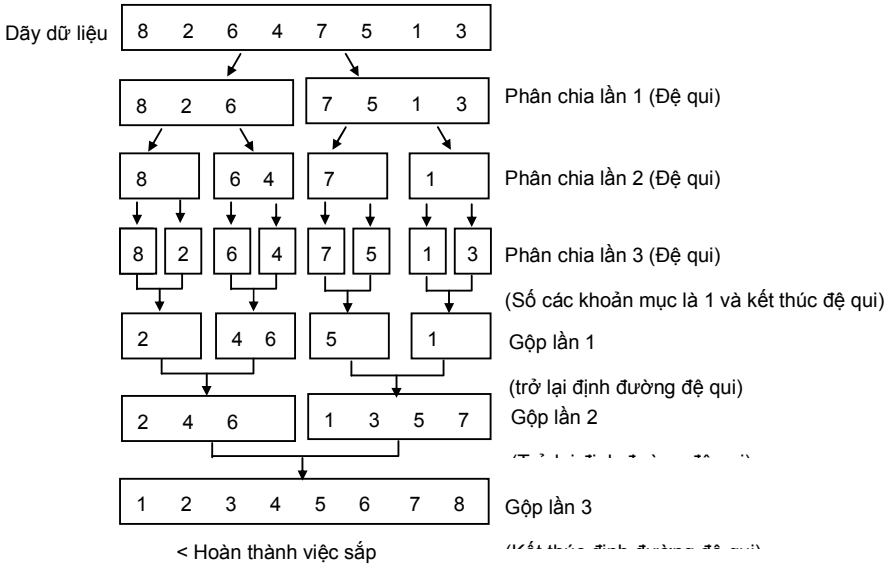
- Lời gọi đệ qui và phương pháp chia và trị được dùng, như trong trường hợp phương pháp sắp xếp nhanh.
- Vì các dãy dữ liệu được truy nhập tuần tự và được sắp xếp, nên thuật toán sắp xếp gộp được dùng cho việc sắp xếp ngoài, chẳng hạn sắp xếp dữ liệu trên băng từ.

<Thủ tục>

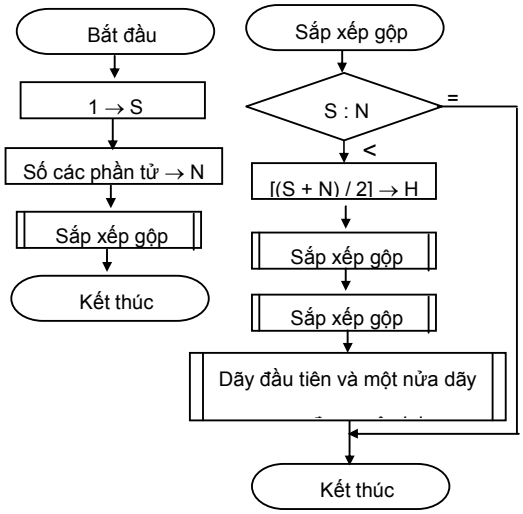
1. Dữ liệu được phân chia thành hai phần và từng phần được chia nhỏ thêm nữa cho tới khi chỉ còn lại một phần từ trong dãy dữ liệu.
2. Sau khi dãy dữ liệu được phân chia, các phần được phân chia được gộp tuần tự lại.

Các hình 2-2-23 và 2-2-24 đưa ra trạng thái của dãy dữ liệu trong các thao tác sắp xếp gộp và lưu đồ của phương pháp sắp xếp gộp.

Hình 2-2-23 Trạng thái của dãy dữ liệu trong các thao tác sắp xếp



Hình 2-2-24 Lưu đồ của phương pháp sắp xếp gộp



Các thông số (tham số)

- Dãy dữ liệu: TBL
- Vị trí bắt đầu trong dãy dữ liệu: S
- Vị trí kết thúc trong dãy dữ liệu: N

Sắp xếp nửa dãy dữ liệu được chia đầu tiên — Các thông số:

- Dãy dữ liệu: TBL
- Vị trí bắt đầu dãy dữ liệu: S
- Vị trí kết thúc dãy dữ liệu: H-1

Sắp xếp nửa dãy dữ liệu — Các thông số

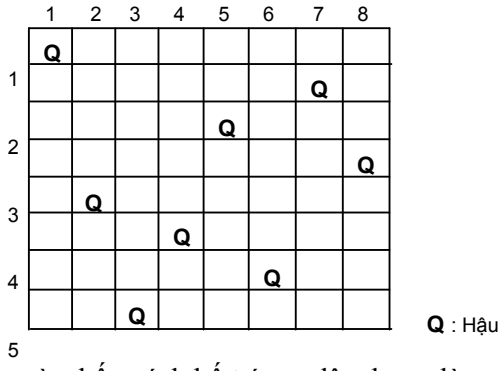
2.2.3 Thuật toán đệ qui

Lời gọi đệ qui là việc một trình gọi tới chính nó trong khi xử lí dữ liệu. Thuật toán được thiết kế bằng việc dùng lời gọi đệ qui là thuật toán gọi đệ qui. Các thuật toán sắp xếp nhanh và sắp xếp gộp cũng là những thuật toán đệ qui.
Mục này xét "bài toán tám hậu" như một thí dụ để giải thích cách thuật toán đệ qui làm việc.

(1) Bài toán tám hậu

Bài toán tám hậu là bài toán tìm cách bố trí tám con hậu trên bàn cờ (kẻ ô 8x8) sao cho không con nào ăn được con nào. Con hậu là một con cờ, và nó có thể ăn con cờ khác nằm trên các đường đứng, ngang hay chéo nối với nó. Do đó trong câu trả lời, các con hậu phải được đặt theo cách chúng không ở vào trên cùng một đường thẳng. Hình 2-2-25 nêu ra một trong những câu trả lời.

Hình 2-2-25 Ví dụ về câu trả lời cho câu hỏi tám hậu



Trong việc giải bài toán này, bốn cách bố trí sau đây được dùng:

q [i]: Vị trí nơi một con hậu được đặt vào cột thứ i (i=1 tới 8)
Trong ví dụ trên, q = {1, 5, 8, 6, 3, 7, 2, 4}

x [j]: Chỉ ra liệu có con hậu nào trên hàng thứ j (j=1 tới 8) không

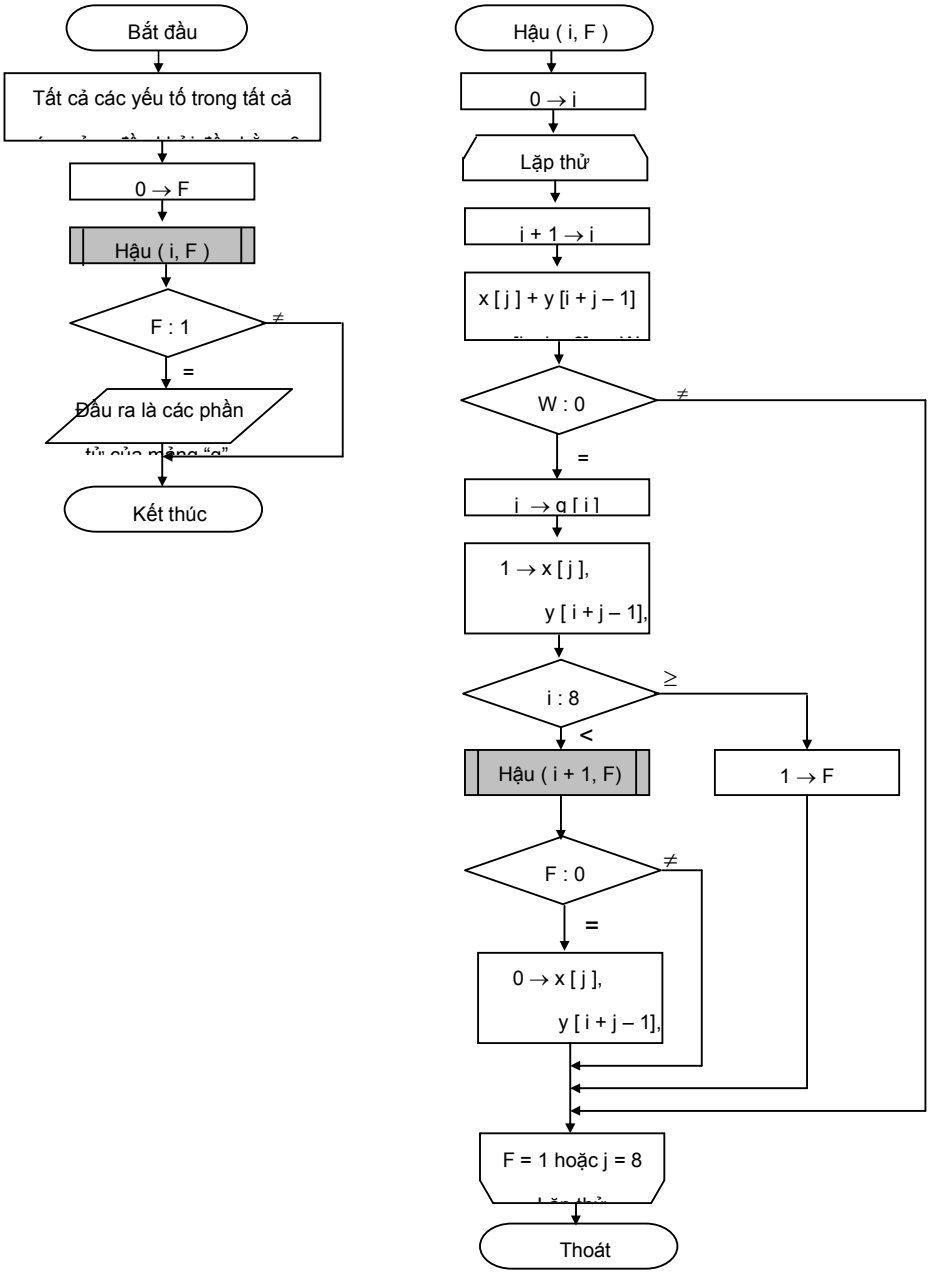
y [k]: Chỉ ra liệu có con hậu nào trên đường chéo đi lên từ trái sang phải thứ k không.
Trên cùng đường chéo đi lên trái sang phải, i + j bao giờ cũng như nhau. Điều này có thể được dùng để thu được một chỉ số "k" với i+j-1 (k=1 tới 15).

Z [l]: Chỉ ra liệu có con hậu nào ở đường chéo xuống dưới từ trái sang phải ở vị trí thứ l không. Trên cùng đường chéo xuống dưới trái sang phải, i-j bao giờ cũng như nhau. Điều này có thể được dùng để lấy chỉ số "l" với i-j+8 (l=1 tới 15).

* Với các mảng x, y và z, các chỉ số 0 và 1 nghĩa là 'không có con hậu' và 'có con hậu' tương ứng.

Hình 2-2-26 đưa ra lưu đồ về thuật toán cho việc giải bài toán này.

Hình 2-2-26 Lưu đồ của bài toán tám hậu



2.2.4 Xử lí xâu kí tự

Kí tự được duyệt, chèn thêm, xoá, tráo đổi hay nén lại. Mục này mô tả việc duyệt và nén xâu.

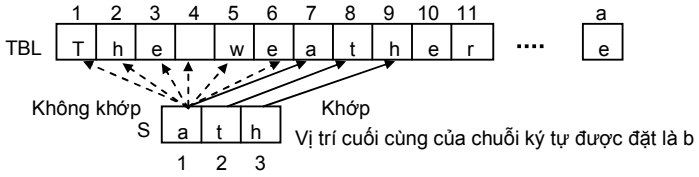
(1) Duyệt xâu kí tự

Để duyệt xâu kí tự, ta dùng thuật toán cho việc duyệt một mẫu xâu nào đó và định vị nó trong xâu kí tự.

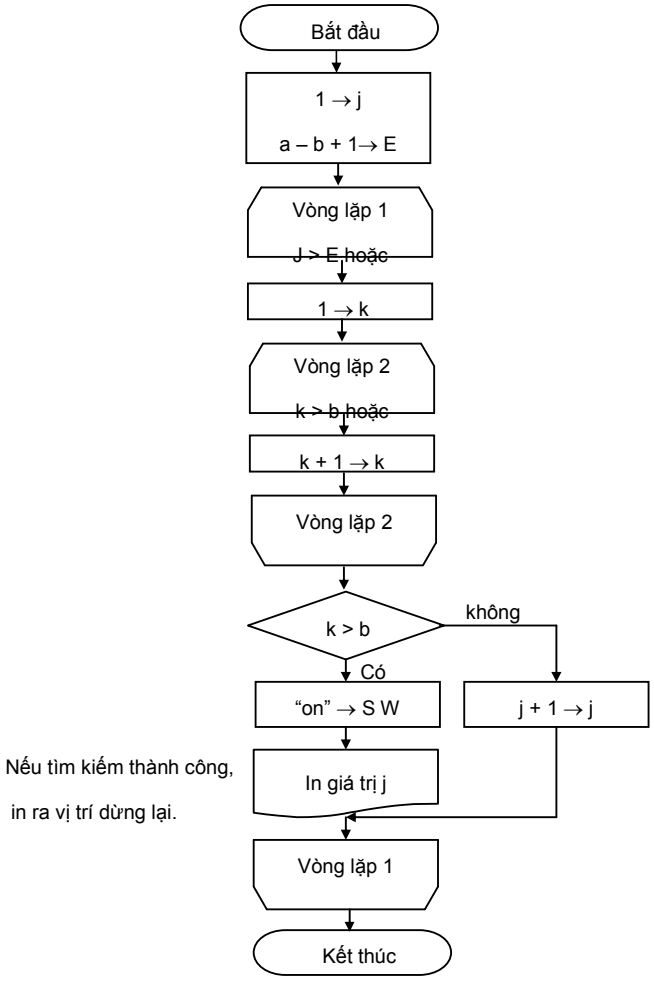
① Đối sánh đơn giản

Văn bản được so sánh theo từng kí tự một cách tuần tự từ đầu. Khi một xâu kí tự được duyệt qua, thì vị trí tại đó đang duyệt qua sẽ được đặt là giá trị cho lại. Về nguyên tắc, việc so sánh này được thực hiện lặp lại cho tới khi kí tự thứ nhất của xâu kí tự cần duyệt sánh đúng với kí tự trong xâu kí tự của văn bản. Nếu có sự sánh đúng, từng kí tự còn lại của xâu kí tự đem sánh sẽ được so sánh với từng kí tự của xâu kí tự cần duyệt.

Hình 2-2-27 Hình ảnh của việc duyệt



Hình 2-2-28 Lưu đồ của đối sánh đơn



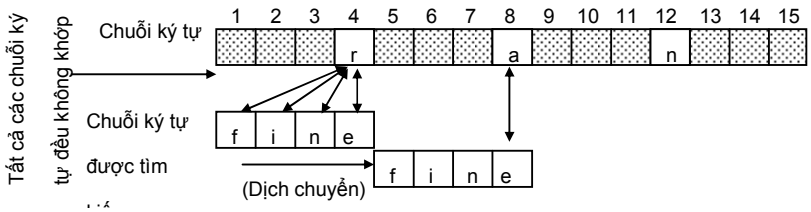
② Phương pháp Boyer-Moore (phương pháp BM)

Trong thuật toán Boyer-Moore, dữ liệu được đối sánh trong khi các kí tự trong văn bản được bỏ qua. Trong mục này, ta chỉ giải thích lược đồ cơ sở của thuật toán này.

a. Nếu không có xâu kí tự nào để duyệt

Hình 2-2-29 chỉ ra việc so sánh đuôi của xâu kí tự cần duyệt và mẫu văn bản.

Hình 2-2-29 Phương pháp BM (trường hợp 1)



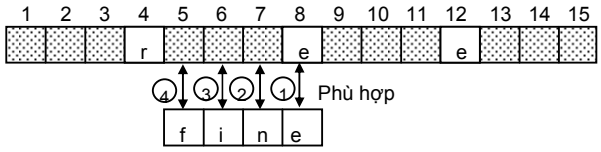
Trong trường hợp này, kí tự tại đuôi, và tất cả các kí tự khác trong phần văn bản thứ nhất không sánh đúng với bất kì kí tự nào của xâu kí tự cần được duyệt. Do đó, điểm duyệt được chuyển đi theo chiều dài của xâu kí tự được duyệt để cho phép việc duyệt

tiếp được bắt đầu từ điểm đó.

b. Nếu có việc sánh đúng với một kí tự tại đuôi của xâu kí tự cần duyệt

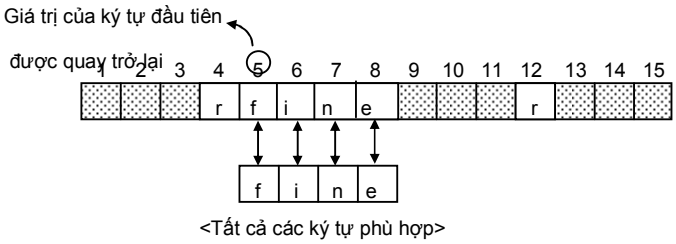
Hình 2-2-30 đưa ra trường hợp trong đó một kí tự tại đuôi của xâu kí tự cần duyệt được so sánh với mẫu văn bản và có việc so sánh đúng.

Hình 2-2-30 Phương pháp BM (trường hợp 2)



Vì một kí tự tại đuôi của xâu kí tự so sánh đúng với một kí tự trong văn bản, nên các kí tự đứng trước kí tự so sánh đúng này phải được so sánh. Nếu tất cả các kí tự đều so sánh đúng, thì giá trị chỉ số của kí tự thứ nhất của mẫu văn bản so sánh đúng được cho lại.

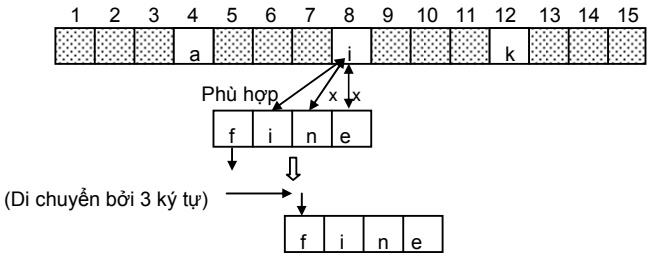
Hình 2-2-31 Duyệt thành công



c. Nếu có việc sánh đúng với một kí tự ở đâu đó trong xâu kí tự nhưng lại không sánh với một kí tự ở đuôi của xâu kí tự

Trong trường hợp này được vẽ ở Hình 2-2-32, xâu kí tự cần duyệt có thể đơn giản được bỏ đi. Vấn đề là khoảng cách di chuyển.

Hình 2-2-32 Phương pháp BM (trường hợp 3)



Khoảng cách di chuyển được xác định bởi cách các kí tự trong xâu kí tự cần duyệt được bố trí như được vẽ trong Hình 2-2-33.

Hình 2-2-33 Khoảng cách di chuyển

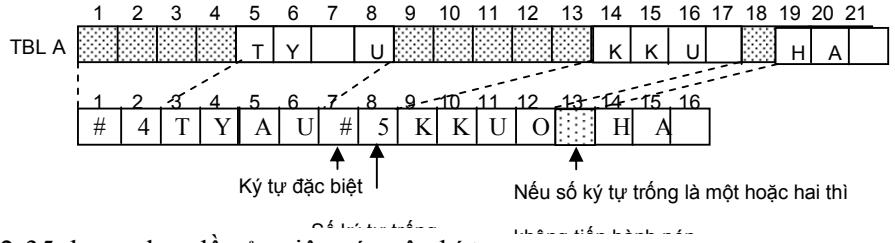
Ký tự	f	i	n	e	các ký tự khác

Bằng cách lưu giữ khoảng cách di chuyển này trong một mảng, xâu kí tự cần duyệt có thể được di chuyển tới vị trí đúng.

(2) Nén xâu kí tự

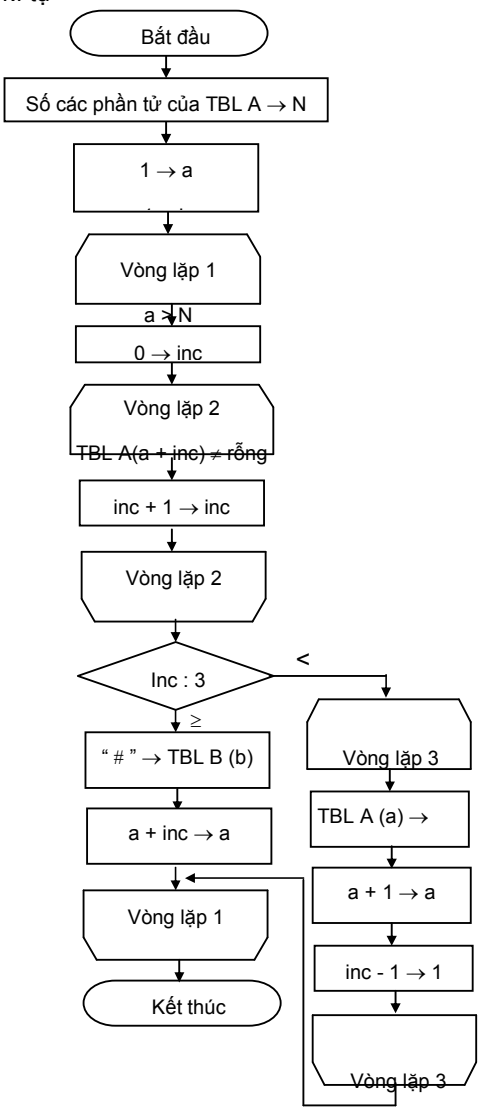
Làm cho một xâu kí tự ngắn lại bằng việc thay thế những kí tự liên tiếp hay dấu cách bằng số lượng kí tự đó được gọi là nén xâu kí tự.
Mục này giải thích thuật toán để nén các kí tự dấu trắng (dấu cách).
Trong trường hợp của ví dụ được nêu trong Hình 2-2-34, xâu kí tự được duyệt từ đầu, và nếu có ba kí tự trắng liên tiếp, chúng sẽ được thay thế bằng một kí tự đặc biệt (#) và số các kí tự trắng.

Hình 2-2-34 Hình ảnh về nén xâu kí tự



Hình 2-2-35 đưa ra lưu đồ của việc nén xâu kí tự.

Hình 2-2-35 Lưu đồ của việc nén xâu kí tự



2.2.5 Xử lí tệp

Các tệp khác nhau được dùng để thực hiện những nhiệm vụ giấy tờ. Đặc trưng của xử lí tệp là xử lí cho từng bản ghi một.
Thuật toán xử lí tệp điển hình là như sau:

Ví dụ
Xử lí chuẩn bị: Mở tệp, xoá bộ đếm v.v..
Xử lí chính: Tính toán, soạn thảo, đưa ra v.v...
Xử lí kết thúc: Đóng tệp, v.v..

Mục này mô tả một thuật toán cho việc xử lí các tệp cùng kiểu và thuật toán khác cho việc xử lí các tệp kiểu khác nhau.

(1) Kiểm soát nhóm trong xử lí các tệp cùng kiểu

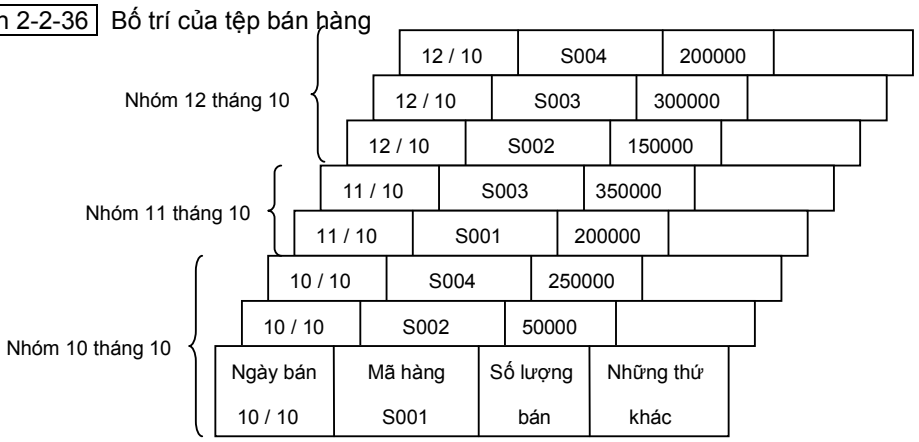
Kiểm soát nhóm (tổng nhóm) là để xử lí các bản ghi với cùng khoá như một tổng thể. Trong việc tính tổng số bán hàng cho từng mã khách hàng (khoá = mã khách hàng) hay mức trung bình cho từng lớp (khoá = mã lớp), chẳng hạn, kiểm soát nhóm là một trình xử lí không thể thiếu được.
Kiểm soát nhóm đòi hỏi rằng các bản ghi được sắp xếp theo từng khoá.

Ví dụ Thuật toán để đọc tệp số bán hàng và in ra bản in chi tiết về bán hàng và tổng số bán cho mỗi ngày

① Bố trí tệp bán hàng

Hình 2-2-36 đưa ra bố trí của tệp bán hàng.

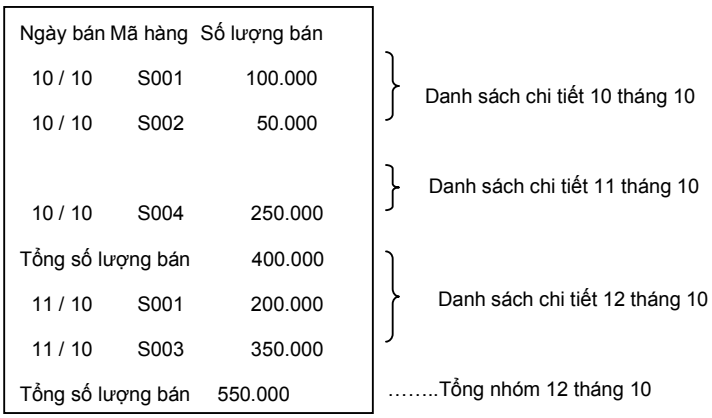
Hình 2-2-36 Bố trí của tệp bán hàng



② Định dạng đưa ra (bản in chi tiết về bán hàng)

Hình 2-2-37 đưa ra định dạng được dùng để in bản in chi tiết về bán hàng và tổng số bán hàng từng ngày.

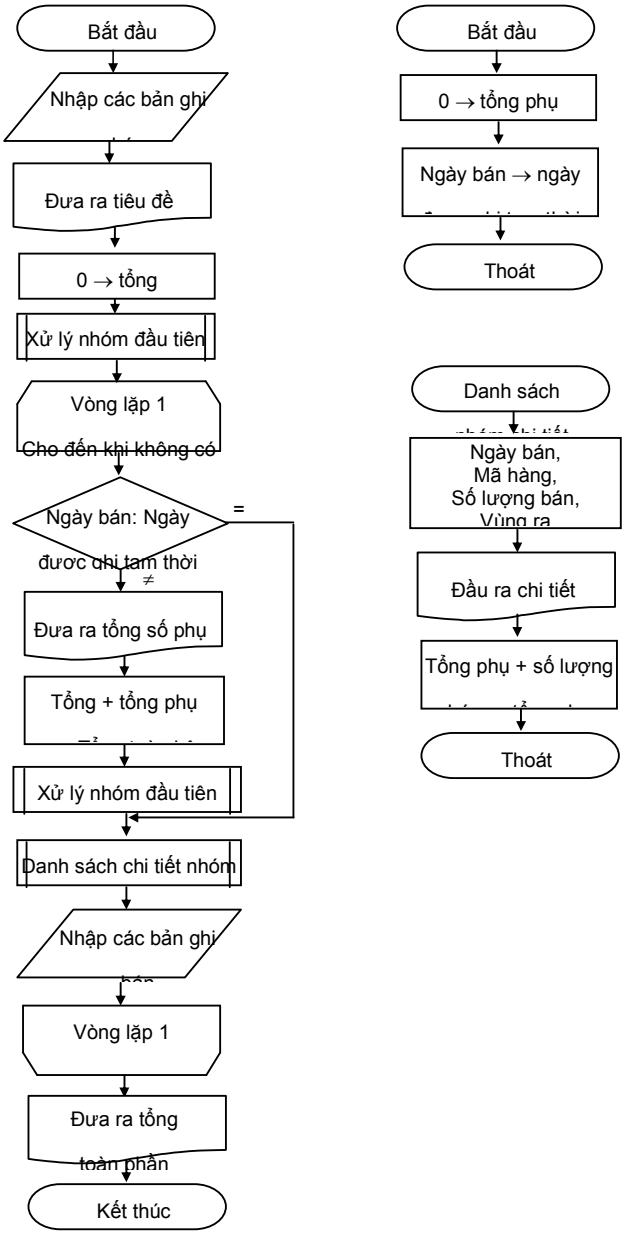
Hình 2-2-37 Định dạng được dùng để đưa ra bản in chi tiết về bán hàng



③ Lưu đồ và sơ đồ chi tiết

Để thu được tổng nhóm, việc phân chia giữa các nhóm phải được phân biệt rõ. Với chủ đề đặc biệt này, điểm mà ngày bán thay đổi trở thành việc phân chia. Do đó, cần xác định liệu ngày bán hàng trong bản ghi mới được nạp vào có sánh đúng với ngày trong các bản ghi đã xử lí gần đây nhất hay không. Để làm điều này, một khoá (ngày bán hàng) tạm thời được cất giữ trước khi bản ghi đầu tiên trong nhóm được xử lí, và nó được so sánh với khoá (ngày bán hàng) của các bản ghi quá khứ.
Hình 2-2-38 đưa ra lưu đồ của kiểm soát nhóm.

Hình 2-2-38 Lưu đồ của kiểm soát nhóm



(2) Cập nhật nhiều tệp

Nếu nhiều tệp được so sánh bằng việc dùng cùng tiêu chuẩn bằng việc sánh, thì mọi tệp phải được sắp theo trình tự của khoá, như trong trường hợp của kiểm soát nhóm.
Các nhiệm vụ xử lí tệp là như sau:

- Gộp tệp
- Đối sánh tệp
- Cập nhật tệp
- Duy trì tệp

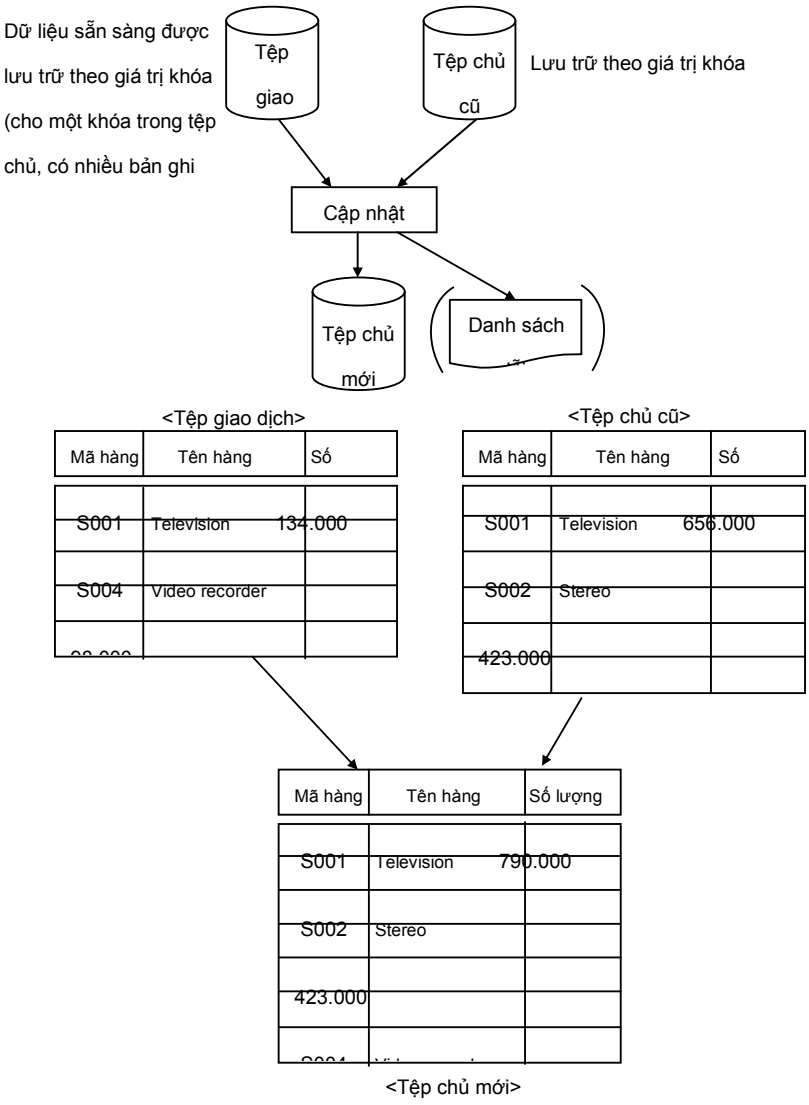
Mục này mô tả nhiệm vụ cập nhật tệp. Việc cập nhật tệp là cập nhật tệp chính dựa trên dữ liệu chứa trong tệp giao tác. Nếu tệp chính là tệp tuần tự, thì nó phải được tạo mới lại hoàn toàn. Nếu tệp chính có thể được truy nhập ngẫu nhiên, thì không cần tạo ra tệp chính mới vì các bản ghi có thể được tìm kiếm bằng các khoá, và được cập nhật. Tại đây, chúng ta giải quyết với thủ tục cập nhật nếu tệp chính là tệp tuần tự.

① Cập nhật 1:1

Cập nhật 1:1 nghĩa là việc cập nhật được thực hiện nếu từng bản ghi trong tệp chính sánh đúng với một bản ghi trong tệp giao tác. (Xem Hình 2-2-39.)
Nhiều tệp được đọc và trình xử lí được xác định bằng việc so sánh kích cỡ của từng khoá như sau:
Kích cỡ của khoá

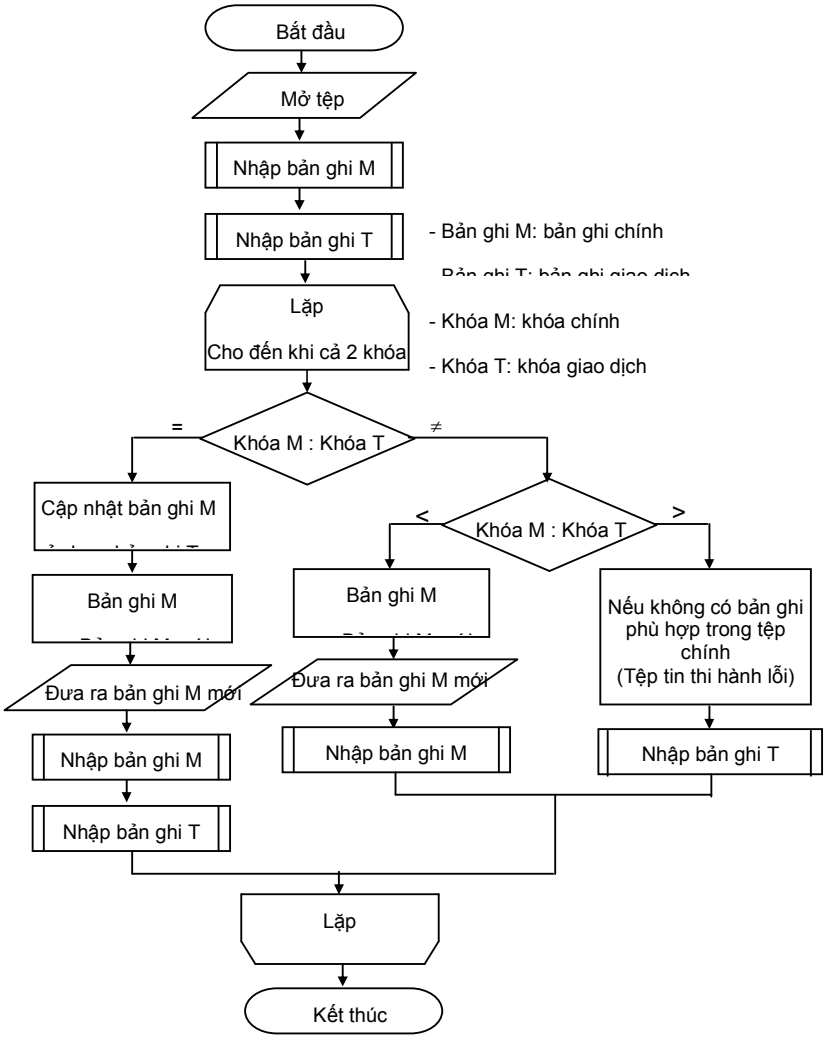
- TR key = MS key Dữ liệu được cập nhật.
- TR key > MS key Dữ liệu được sao (dữ liệu được viết vào tệp chính mới).
- TR key < MS key Dữ liệu được bổ sung vào tệp chính mới (có thể là trường hợp mà trạng thái này được xem như lỗi và trình xử lí lỗi này thực hiện).

Hình 2-2-39 Hình ảnh về cập nhật 1:1



Hình 2-2-40 đưa ra lưu đồ của cập nhật 1:1.

Hình 2-2-40 Lưu đồ của cập nhật 1:1

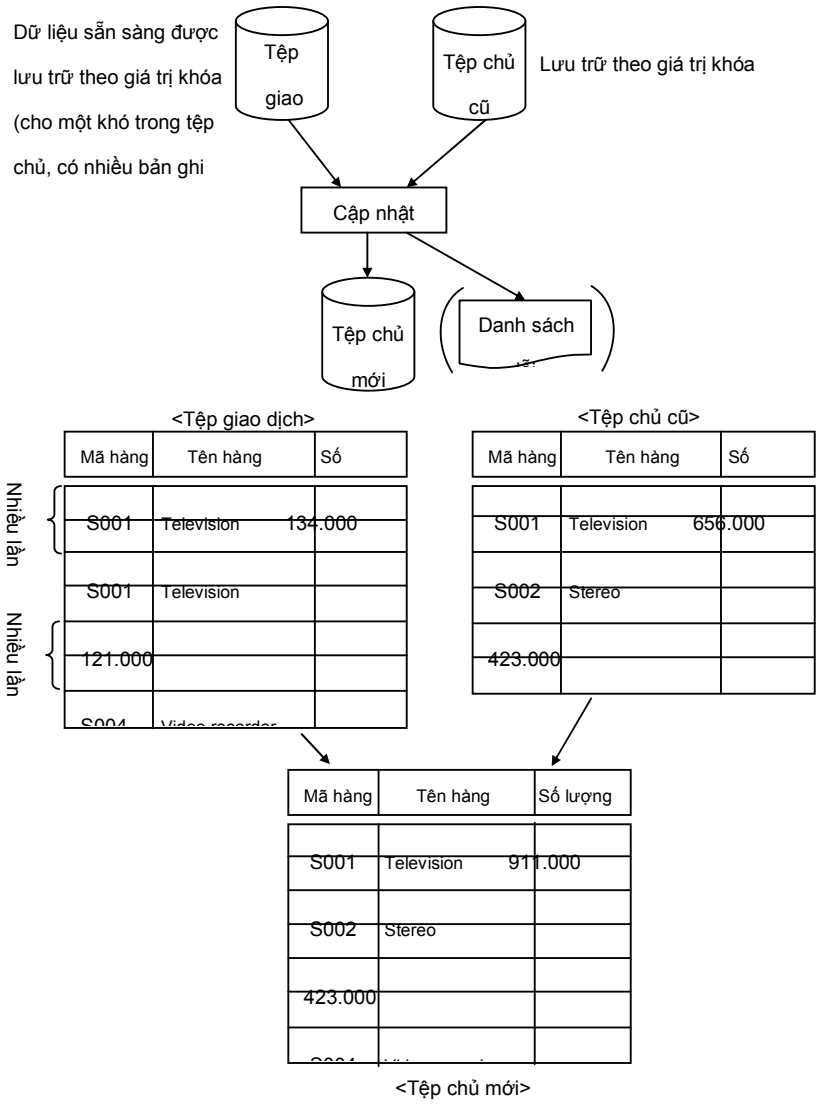


② Cập nhật 1:n

Cập nhật 1:n là trình cập nhật được dùng nếu một bản ghi trong tệp chính sánh đúng với nhiều bản ghi trong tệp giao tác. (Xem Hình 2-2-41.)
Về nguyên tắc, nhiều tệp được đọc vào, như trong trường hợp của cập nhật 1:1, và các trình xử lí được xác định bằng cách so sánh kích cỡ của khoá như sau:
Kích cỡ của khoá

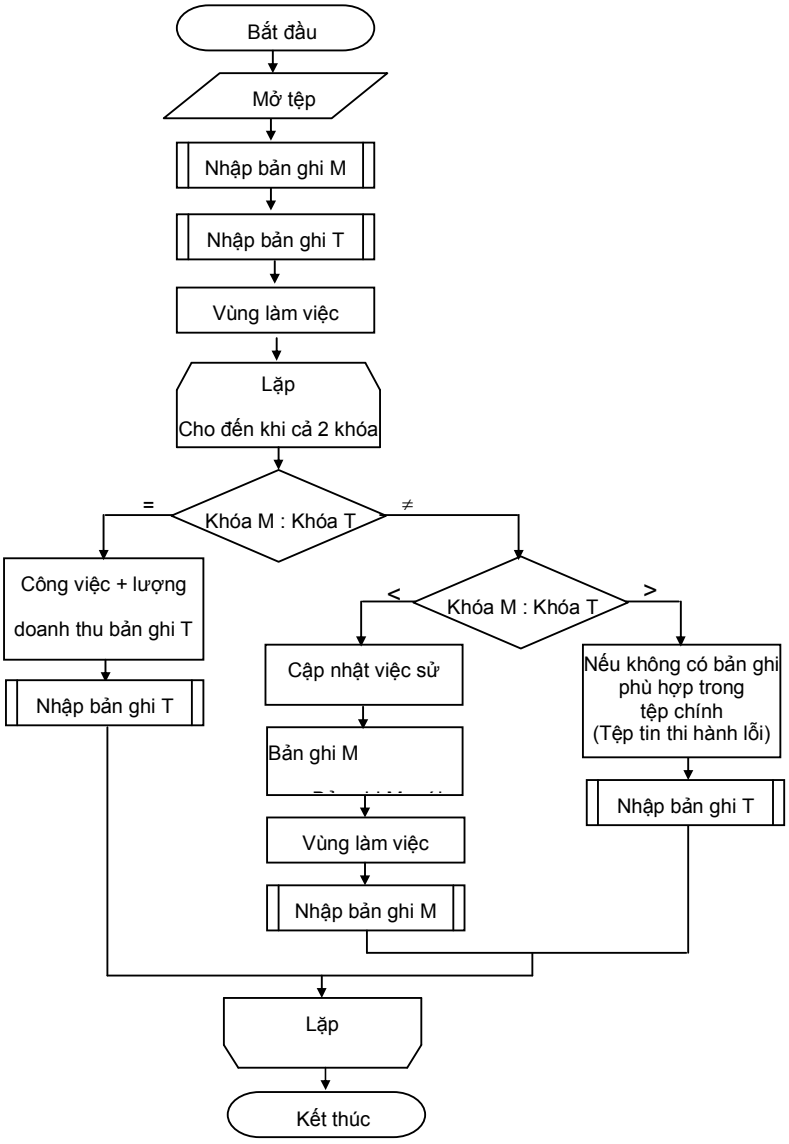
- TR key = MS key Dữ liệu được lấy tổng và kết quả được lưu giữ trong vùng làm việc.
- TR key > MS key Dữ liệu được cập nhật.
- TR key < MS key Dữ liệu được bổ sung vào tệp chính mới (có thể có trường hợp điều này được coi là lỗi và trình xử lí lỗi được gọi tới).

Hình 2-2-41 Hình ảnh về cập nhật 1:n



Hình 2-2-42 đưa ra lưu đồ của cập nhật 1:n.

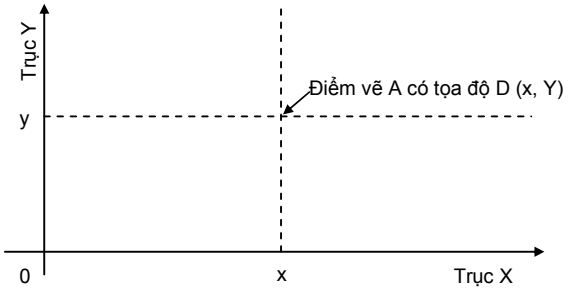
Hình 2-2-42 Lưu đồ của cập nhật 1:n



2.2.6 Vẽ hình

Trong thế giới máy tính hiện tại, CAD, CG và những công nghệ vẽ hình đang được sử dụng. Trong thuật toán vẽ hình, một hình được đối xử như một tập các chấm. Cách một đường thẳng đơn và đường tròn được vẽ ra sẽ được giải thích trong mục này.
Hàm D (x, y) được dùng để vẽ một chấm tại điểm giao nơi đường theo toạ độ x và đường theo toạ độ y gặp nhau, như được đưa ra trong Hình 2-2-43.

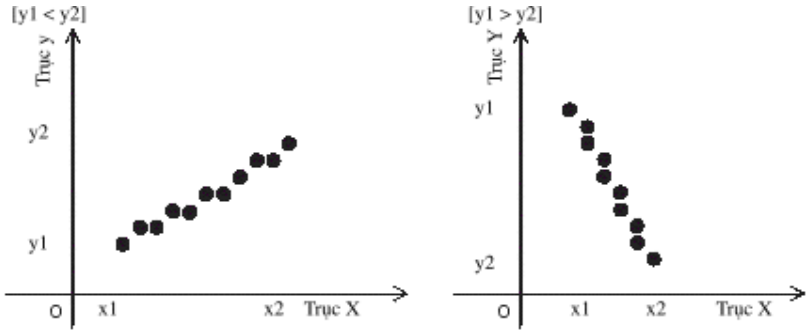
Hình 2-2-43 Cách hàm D (x, y) làm việc



(1) Vẽ đường thẳng

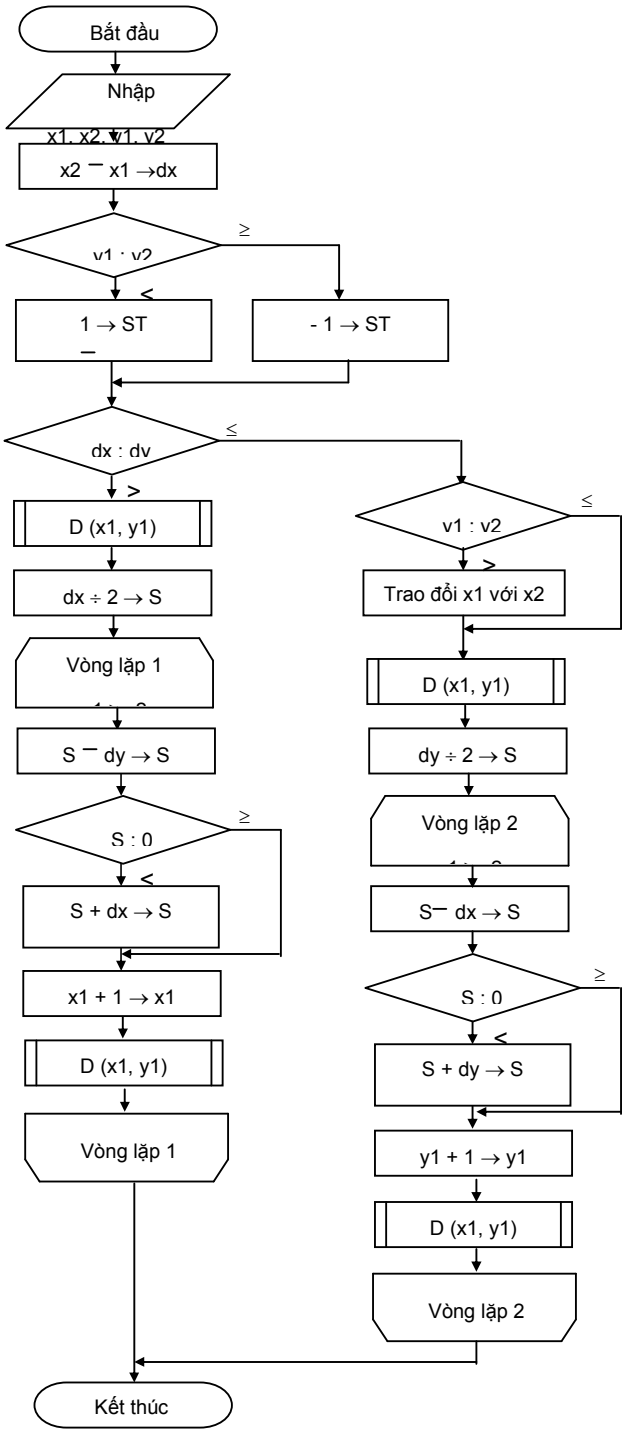
Hình 2-2-44 nêu việc vẽ đường thẳng.

Hình 2-2-44 Ví dụ về việc vẽ đường thẳng



Lưu đồ được đưa ra trong Hình 2-2-45 là một thuật toán để vẽ đường thẳng nối hai điểm đã cho theo toạ độ, tức là, (x1, y1), (x2, y2) $\left| 0<x1\leq x2,\ 0<y1,\ 0<y2 \right|$. Mặc dầu thuật toán này có thể vẽ đường thẳng chạy xiên theo hướng trên cao hay dưới thấp bằng việc vẽ các chấm theo toạ độ nguyên, nó được thiết kế để cho phép các đường trông như đường thẳng. Cũng vậy, các phép toán nhân chia được giữ tới mức tối thiểu còn phép toán cộng-trừ được dùng nhiều nhất để làm tăng tốc độ xử lí.

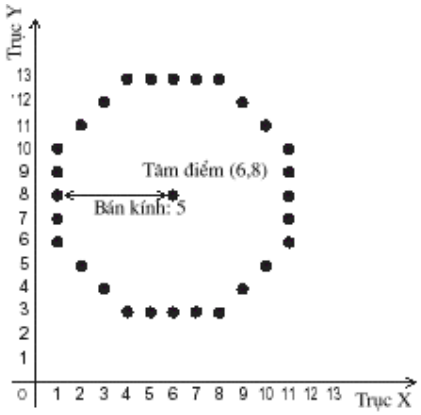
Hình 2-2-45 Lưu đồ của việc vẽ đường thẳng



(2) Vẽ đường tròn

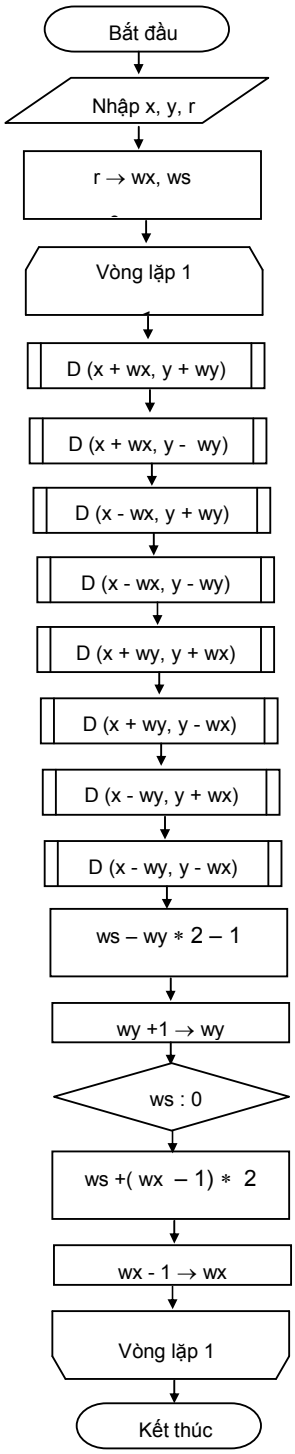
Hình 2-2-46 nêu cách vẽ đường tròn.

Hình 2-2-46 Cách vẽ đường tròn



Lưu đồ trong Hình 2-2-47 nêu ra thuật toán vẽ đường tròn dựa trên toạ độ (x, y) của tâm điểm và bán kính r. Để vẽ đường tròn này, không nên dùng hàm lượng giác cần thời gian thực hiện lâu.

Hình 2-2-47 Lưu đồ cho việc vẽ đường tròn



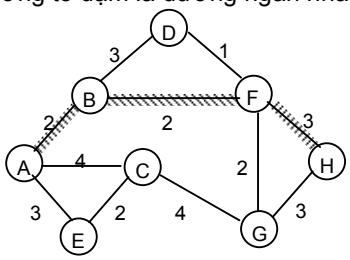
2.2.7 Đồ thị

Đồ thị chúng ta thảo luận trong mục này là đồ thị được tạo nên từ các cung nối nhiều điểm. Về cơ bản ta giả thiết là đồ thị vô hướng. Đồ thị có hướng cũng có thể được dùng, tuỳ theo kiểu vấn đề cần được giải quyết.
Bài toán đường đi ngắn nhất được mô tả ở đây như một trong những bài toán đồ thị tiêu biểu.

(1) Bài toán đường đi ngắn nhất

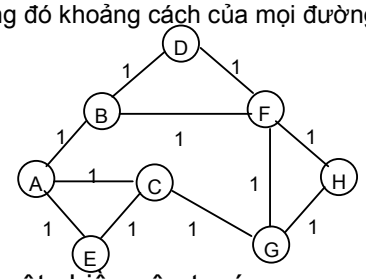
Như được nêu trong Hình 2-2-48, có các đường và nút, và phải tìm ra đường đi ngắn nhất từ điểm A tới điểm H. Bởi vì con số dọc theo mỗi đường chỉ ra khoảng cách, nên bạn có thể thấy ngay rằng đường tô đậm là đường ngắn nhất.

Hình 2-2-48 Bản đồ (đường tô đậm là đường ngắn nhất)



Đồ thị được vẽ trên Hình 2-2-48 có các con số dọc theo từng đường để chỉ ra khoảng cách đường được gọi là đồ thị có trọng số. Mặc dầu chúng ta có thể thấy rõ đường ngắn nhất bằng mắt, thuật toán phải được dùng để làm cho máy tính tìm ra đường ngắn nhất qua tính toán.
Chúng ta sẽ mô tả ở đây ba phương pháp tìm đường: phương pháp chiều sâu trước và chiều rộng trước, vốn đảm bảo rằng khoảng cách của mỗi đường là 1, và phương pháp duyệt của Dijkstra vốn là phương pháp duyệt chính để tìm ra đường đi ngắn nhất.

Hình 2-2-49 Đồ thị trong đó khoảng cách của mọi đường đều được đặt là 1



① Phương pháp duyệt chiều sâu trước

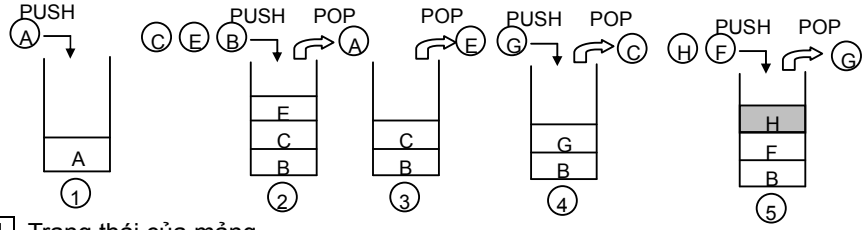
Dùng phương pháp duyệt chiều sâu trước, một đường được chọn làm điểm bắt đầu và việc duyệt bắt đầu tìm con đường tới đích bằng việc dùng chồng.

<Thủ tục> (Hình 2-2-50)

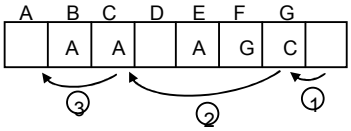
1. Một nút tại điểm bắt đầu được đặt vào chồng.
2. Một nút được lấy ra từ chồng. Các nút kề với nút vừa lấy ra được kiểm tra và những nút trước đây chưa bao giờ được nạp vào chồng được chọn lấy. Các nút được chọn được nạp vào chồng.
3. Khi các nút được chọn được nạp vào chồng, thì nút được lấy ra ở bước 2 trên được lưu giữ vào trong mảng.

4. Các bước 1, 2 và 3 được thực hiện lặp lại cho tới khi nút đích được đặt vào chồng hay chồng trở thành rỗng.

Hình 2-2-50 Trạng thái của ngăn xếp



Hình 2-2-51 Trạng thái của mảng



Trước khi các nút được chọn được đưa vào trong ngăn xếp, nút đã chọn ở bước 2 ở trên phải được lưu trữ trong mảng. Ví dụ nếu A được chọn và B, C, và E được đưa vào trong ngăn xếp. A phải được lưu trữ trong các phần tử của B, C và E. Sau khi nút tìm ra được đặt vào trong ngăn xếp, các nút còn lại được tìm lần lượt để tìm ra đường ngắn nhất. (đường tương tự được thực hiện trong

<Kết quả>

Việc duyệt được thực hiện bằng phương pháp chiều sâu trước cho kết quả trong đường A-C-G-H được vẽ trong Hình 2-2-51. Mặc dầu hình trên vẽ ra sơ đồ cơ sở của việc thực hiện duyệt, điều xảy ra trong việc duyệt thực tế còn phức tạp hơn. Tức là, giả sử rằng đường ngắn nhất không thể tìm được, các đường khác được duyệt lặp lại, tức là khi việc duyệt đạt tới bước trước bước đặt H, được vẽ trong Hình 2-2-50, bước ⑤ trong chồng, nó nhảy lùi về bước 1 và lặp lại tất cả các bước. Đoạn trình này được lặp cho tới khi chồng trở thành rỗng để cho tất cả các đường đều được tính tới để tìm ra đường ngắn nhất.

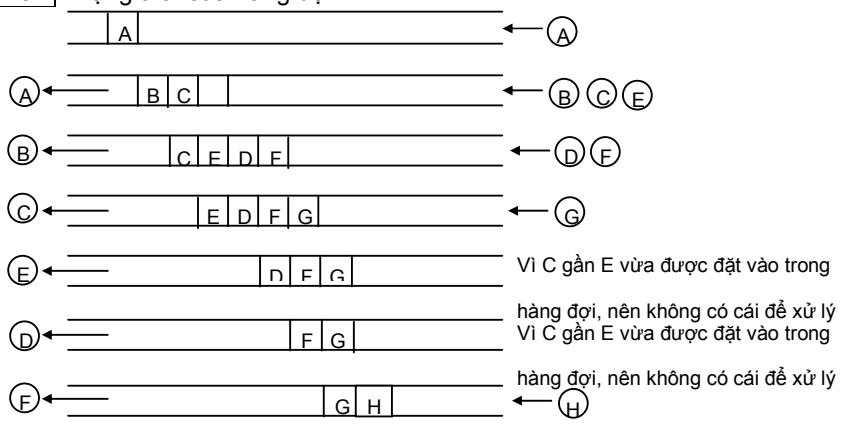
② Phương pháp duyệt chiều rộng trước

Dùng phương pháp duyệt chiều rộng trước, tất cả các đường có thể dẫn tới đích đều được duyệt qua bằng việc dùng hàng đợi.

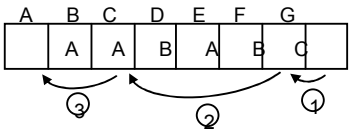
<Thủ tục> (Hình 2-2-52)

1. Một nút tại điểm bắt đầu được đặt vào trong hàng đợi.
2. Một nút được lấy ra khỏi hàng đợi. Các nút kề với nút được lấy ra này được kiểm tra còn những nút chưa bao giờ được đặt vào trong hàng đợi thì được chọn. Các nút được chọn được đặt vào hàng đợi.
3. Nút được lấy ra trong bước 2 được lưu giữ vào mảng.
4. Các bước 1, 2 và 3 trên được thực hiện lặp lại cho tới khi một nút đích được đạt tới hay khi hàng đợi trở thành rỗng.

Hình 2-2-52 Trạng thái của hàng đợi



Hình 2-2-53 Trạng thái của mảng



Trước khi nút được đặt vào trong hàng đợi, nút tìm ra ở bước 2 được lưu trữ trong mảng. Ví dụ, nếu a được tìm ra và B, C được đặt vào trong

Việc duyệt được thực hiện bằng việc dùng phương pháp duyệt chiều rộng trước cho kết quả trong đường đi ngắn nhất A-B-F-H được vẽ trong Hình 2-2-53. Tất cả các nút đều được duyệt lần lượt, như được nêu ở trên, trong khi các tính toán được thực hiện để tìm đường tốt nhất.

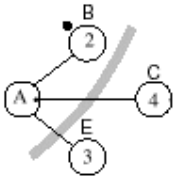
③ Phương pháp duyệt của Dijkstra

Phương pháp duyệt của Dijkstra là phương pháp áp dụng cách duyệt chiều rộng trước.

<Thủ tục> (Hình 2-2-54)

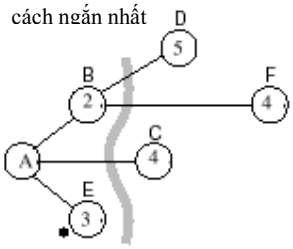
1. Khoảng cách tới từng nút kề với nút bắt đầu được đo và nút nằm tại khoảng cách ngắn nhất sẽ được chọn (nút này được gọi là X).
2. Khoảng cách tới từng nút kề với nút X từ nút bắt đầu cũng như khoảnh cách tới các nút ngoại trừ X từ nút bắt đầu là được đo và nút nằm ở khoảng cách ngắn nhất được chọn.
3. Bước 2 được thực hiện lặp lại trên mọi nút cho tới khi đạt tới nút đích.
4. Bằng việc bổ sung thêm khoảng cách được gắn với từng nút, có thể thiết lập ra con đường ngắn nhất.

Hình 2-2-54 Phương pháp duyệt của Dijkstra

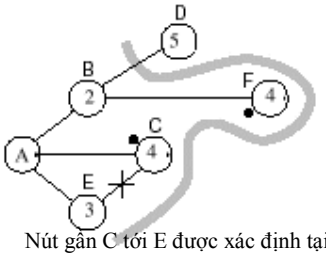


Đường tô đậm là đường chia các nút riêng hoặc các tuyến đường

So sánh nút bắt đầu và nút cạnh nó và đánh dấu nút xác định khoảng cách ngắn nhất

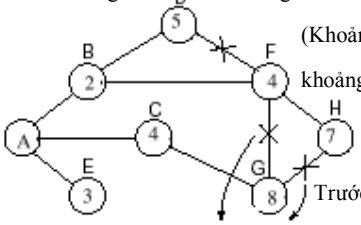


So sánh khoảng cách tới nút xác định tại tuyến đường A tới B với khoảng cách giữa nút liền kế tới A và nút được đánh dấu khoảng cách ngắn nhất



Nút gần C tới E được xác định tại khoảng cách ngắn nhất có thể được duyệt qua A.

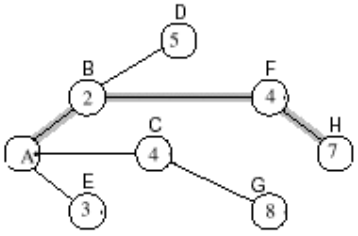
Vì khoảng cách từ A tới C ngắn hơn khoảng cách từ A tới C qua E



(Khoảng cách đường từ B tới F ngắn hơn khoảng cách đường từ B tới F qua D. Vì

Trước khi tuyến đường bị bỏ, xem xét khoảng cách

Cả hai đường đều bị cắt bỏ



Kết quả thu được tương tự như phương pháp duyệt chính xác

<Tìm kiếm xong>

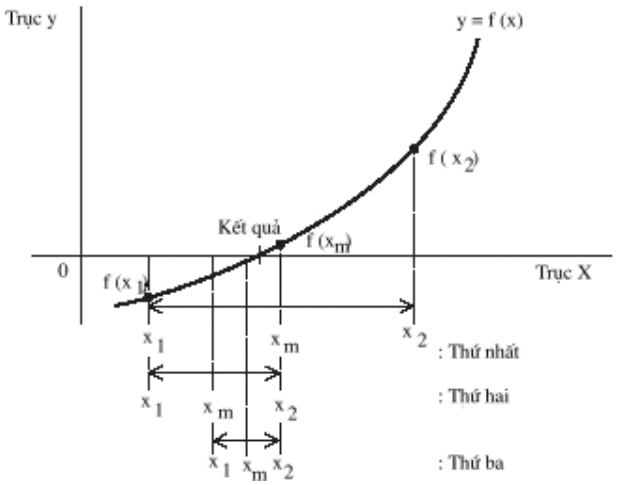
2.2.8 Tính toán số

(1) Nghiệm phương trình đại số

① Phương pháp phân đôi

Phương pháp phân đôi là giải pháp đơn giản nhất của phương trình bậc cao. Phương trình bậc cao thông thường được định nghĩa là y = f(x) và các giá trị khác nhau được gán cho x. Đường được vẽ ra bằng cách gán các giá trị cho x. Giá trị của x khi đường này giao với y = 0 (trục x) trở thành nghiệm của phương trình này.
Hình 2-2-55 là đồ thị được vẽ dựa trên y = f(x).

Hình 2-2-55 Đồ thị được vẽ dựa trên y = f(x)



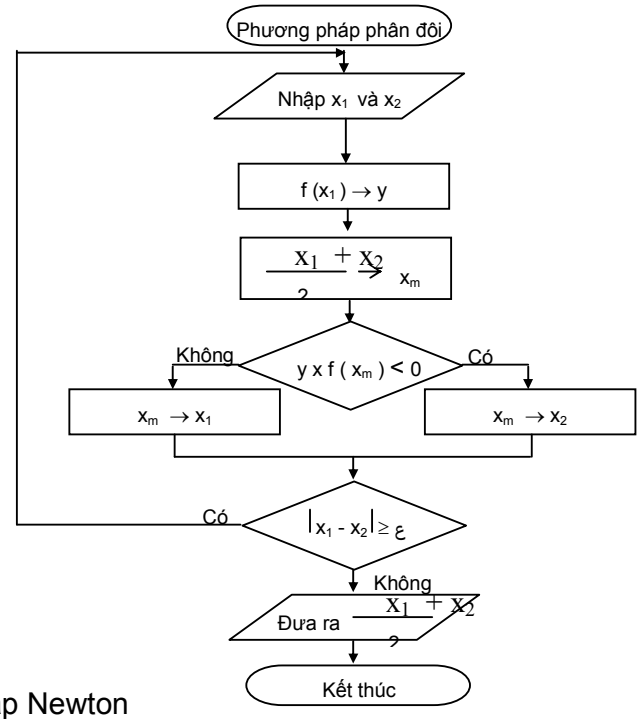
$f(x_1) \cdot f(x_2)<0$ chỉ ra rằng có ít nhất một giải pháp bên trong vùng $[x_1, x_2]$ trong đó đường này giao với trục x. Bằng việc dùng phương pháp phân đôi, vùng $[x_1, x_2]$ này được chia thành hai phần. Để thu được lời giải xấp xỉ, tiến trình phân chia một miền thành hai phần được thực hiện lặp đi lặp lại để chọn ra phần (được chỉ bởi $\leftrightarrow$) có lời giải.

Hình 2-2-56 Thuật toán của phương pháp phân đôi

Bước 1 Tính xm chia vùng [x1, x2] thành 2 phần [x1, xm] và [xm, x2]
Bước 2 Mỗi phần được kiểm tra để xác định xem phần nào có lời giải. Nếu f(x1).f(x2) < 0, thì phần có lời giải ở bên trái, ngược lại phần có lời giải ở bên phải.
Bước 3 Nếu f(x1) . f(xm) < 0, xm → x2, Ngược lại xm → x1.
Bước 4 So sánh $\lvert x_1 - x_2 \rvert$ và ε để xét xem có đáp án gần đúng hay không. Nếu $\lvert x_1 - x_2 \rvert > \varepsilon$, quay

Hình 2-2-57 đưa ra lưu đồ của thuật toán để thu được lời giải của f(x) = 0 bằng việc dùng phương pháp phân đôi.

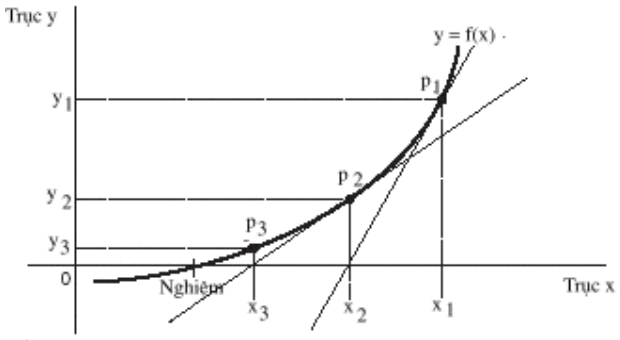
Hình 2-2-57 Lưu đồ của thuật toán để thu được lời giải của f(x) = 0 bằng việc dùng phương pháp phân đôi



② Phương pháp Newton

Phương pháp Newton giả thiết rằng một lời giải xấp xỉ của phương trình bậc cao là đã được biết. Lời giải xấp xỉ đó được hiệu chỉnh lặp lại để thu được lời giải đúng. Phương pháp của Newton là cao cấp hơn các phương pháp khác ở chỗ tốc độ hội tụ nhanh hơn và có thể thu được cả nghiệm thực và ảo.
Hình 2-2-58 là đồ thị đươc vẽ dùng phương pháp Newton.

Hình 2-2-58 Phương pháp Newton



Hình 2-2-59 Thuật toán của phương pháp Newton

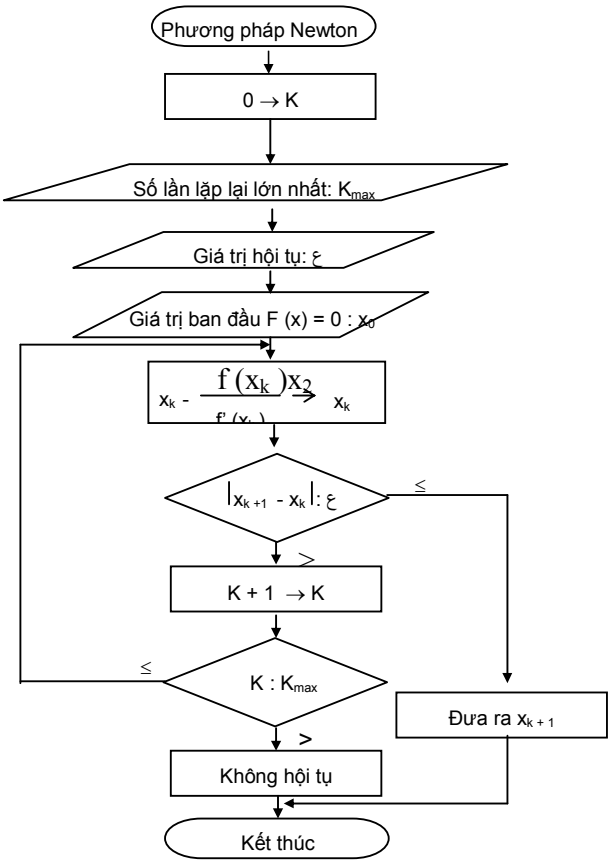
- Bước 1: Tiếp tuyến với y = f(x) tại điểm p_1(x_1, y_1) và cắt trục hoành tại x_2
- Bước 2: Tiếp tuyến với y = f(x) tại điểm p_2 (x_2, y_2). Ở bước này đường tiếp tuyến được thực hiện lặp lại gần với kết quả cần tìm.
- Bước 3: Sự khác nhau giữa các giá trị xấp xỉ nhau trong bước 2 được so sánh với độ chính xác của giá trị hội tụ cho trước. Lặp lại bước 2 và bước 3 cho đến khi sự khác nhau này nhỏ hơn giá trị hội tụ cho trước.

Vì phương trình cho đường tiếp tuyến tại điểm p1 là $y-f(x_1) = f'(x_1)(x-x_1)$, điểm x_2 nơi đường tiếp tuyến giao với trục x có thể thu được bằng việc dùng phương trình sau:

$$x_{i+1} = x_1 - \frac{f(x_i)}{f'(x_i)} \quad (i = 0, 1, 2, ...)$$

Hình 2-2-60 đưa ra lưu đồ của thuật toán để thu được lời giải của f(x) = 0 bằng việc dùng phương pháp Newton.

Hình 2-2-60 Lưu đồ của thuật toán để thu được lời giải của f(x) = 0 bằng việc dùng phương pháp Newton



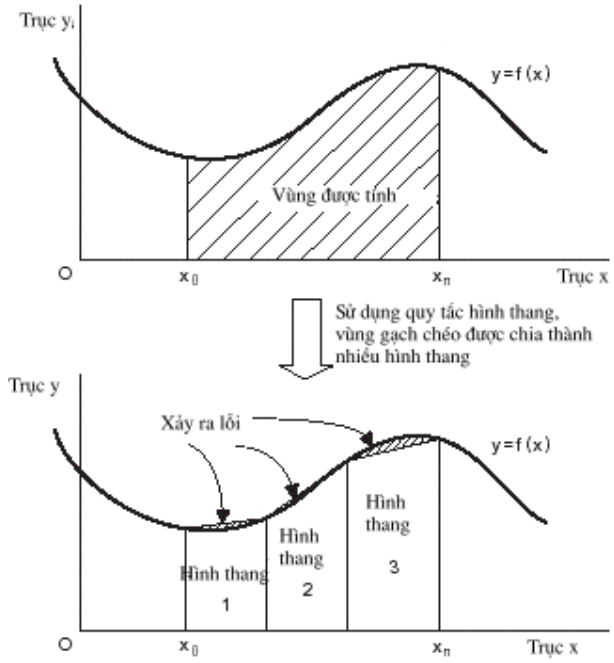
(2) Tích phân số

Phương pháp tích phân số được dùng để tính diện tích của một hình được bao bởi các đường cong. Mục này mô tả qui tắc hình thang và phương pháp Simpson của tất cả các phương pháp tích phân số thông dụng.

① Qui tắc hình thang

Hình 2-2-61 nêu ra khái niệm cơ sở của qui tắc hình thang.

Hình 2-2-61 Khái niệm cơ sở của qui tắc hình thang



Hình 2-2-62 nêu ra thủ tục để tính diện tích được bao bởi đường cong y = f(x), diện tích dọc theo trục x và phần được bao bởi trục x.

Hình 2-2-62 Thủ tục để tính diện tích dùng qui tắc hình thang

Bước 1: Chia đường cong thành các khoảng đều nhau.

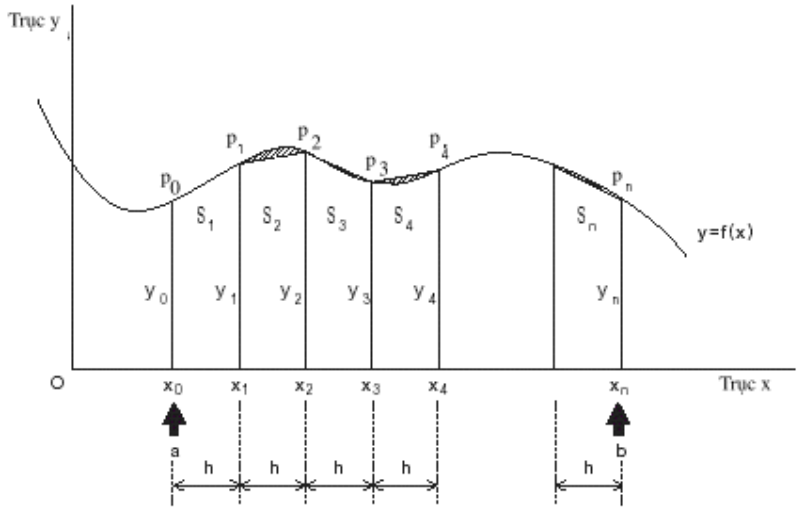
Bước 2: Giả sử đường cong được chia thành các đoạn thẳng, các phần được chia là hình thang.

Bước 3: Mỗi phần được chia được coi là hình thang và tính diện tích hình thang.

Diện tích mỗi hình thang = (cạnh trên + cạnh dưới) x chiều cao ÷ 2

Nếu một diện tích được tính toán bằng việc dùng qui tắc hình thang, thì sai số xuất hiện đối với phần tối, như được vẽ trong Hình 2-2-61, vì diện tích đúng khác với diện tích hình thang. Để làm giảm sai số, phải tăng số các hình thang lên. Hình 2-2-63 đưa ra cách một diện tích được chia thành các hình thang.

Hình 2-2-63 Diện tích được chia ra tương ứng với qui tắc hình thang



Hình 2-2-64 nêu thuật toán để tính diện tích bằng việc dùng qui tắc hình thang dựa trên Hình 2-2-63.

Hình 2-2-64 Thuật toán của qui tắc hình thang

Bước 1: Phần cần tính tích phân, ví dụ [a, b] của y = f(x) được chia thành n phần.

Bước 2: Chia đoạn [a, b] thành n phần bằng nhau, tại mỗi điểm dựng đường vuông góc với trục x.

Bước 3: Giá trị của hàm tại x_0, x_1, x_2, x_3, ...x_n là y_0, y_1, y_2, ...y_n.

Bước 4: Các đường thẳng ở bước 2 cắt đường cong tại p_0, p_1, p_2,…p_n. Với các điểm x, y và p hình thành một hình thang.

Bước 5: Tính diện tích của các hình thang S_1, S_2, ….S_n theo công thức sau:

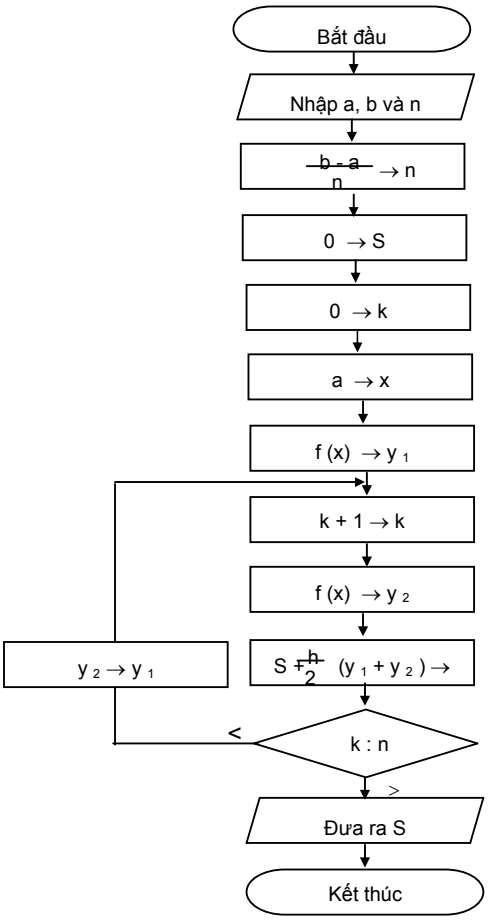
$S_1 = h(y_0 + y_1)/2$

$S_1 = h(y_1 + y_2)/2$

.

Hình 2-2-65 nêu ra lưu đồ của thuật toán để tính diện tích bằng việc dùng máy tính và qui tắc hình thang.

Hình 2-2-65 Lưu đồ của việc tính diện tích xấp xỉ bằng việc dùng qui tắc hình thang

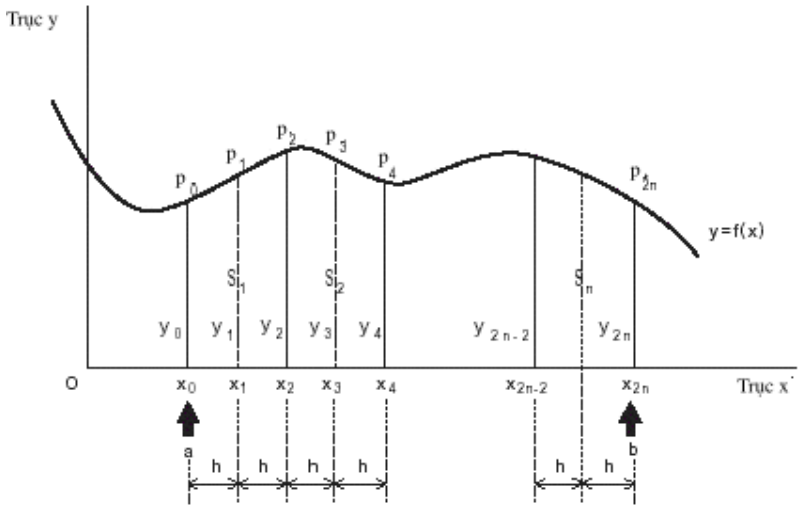


② Phương pháp Simpson

Bằng việc dùng qui tắc hình thang, một đường cong được chia thành các khoảng đều nhau và các điểm giao của đường cong được nối lại để tạo thành các hình thang.Tổng diện tích được tính bằng tổng diện tích của các hình thang. Mặc dầu sai số có thể được giảm đi bằng việc tăng số các hình thang, số các hình thang càng lớn thì thời gian tính toán càng lâu. Hơn nữa, bởi vì phương pháp này dựa trên sơ đồ trong đó một diện tích được bao bởi ba đường thẳng và một đường cong được coi là hình thang, nên có mối quan tâm về độ chính xác của kết quả thu được.

Xem như một giải pháp, phương pháp Simpson được dùng như được nêu trong Hình 2-2-66. Bằng việc dùng phương pháp này, đường cong được làm xấp xỉ thành một parabol dễ xử lí để tính diện tích.

Hình 2-2-66 Diện tích được chia ra bằng việc dùng phương pháp Simpson



Để tính diện tích S1 được bao bởi y = f(x), x = x_0, x = x_2 và trục x được vẽ trong Hình 2-2-66 bằng việc dùng phương pháp Simpson, f(x) được xem như một parabola chạy qua p_0, p_1 và p_2, và S_1 được tính xấp xỉ như sau:

$$\mathbf{S}_1 = \frac{(\mathbf{y}_0 + 4\mathbf{y}_1 + \mathbf{y}_2)\mathbf{h}}{3}$$

Phương pháp này hoàn toàn khác với qui tắc hình thang trong đó một diện tích được chia đều thành 2n phần (phân chia bằng nhau, theo số chẵn), không theo n.
Hình 2-2-67 nêu ra thuật toán để tính diện tích được vẽ trong Hình 2-2-66 bằng việc dùng phương pháp Simpson.

Hình 2-2-67 Thuật toán tính diện tích bằng việc dùng phương pháp Simpson

Bước 1 Đoạn [a, b] nơi lấy tích phân của hàm y = f(x) được chia thành 2n phần bằng nhau.

Bước 2 Các đường thẳng vuông góc với trục x cắt trục x tại các điểm x_0, x_1, x_3, ...x_{2n}

Bước 3 Giá trị của hàm tại các điểm trên là y_0, y_1, …..y_{2n}.

Bước 4 Các đường thẳng trên cắt đường y = f(x) tại các điểm p_0, p_1, ….p_{2n}.

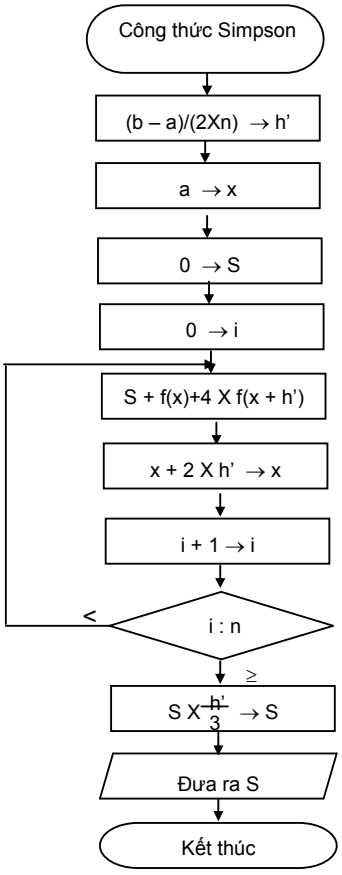
Bước 5 Ba điểm $p_{2i-2}(x_{2i-2}, y_{2i-2})$, p_{2i-1} (x_{2i-1}, y_{2i-1}) và p_{2i} (x_{2i}, y_{2i}) trong hai phần được tính gần đúng như diện tích hình S_i.

$S_1 = (y_0 + 4\ y_1 + y_2)\ h/3$

$S_1 = (y_2 + 4\ y_3 + y_4)\ h/3$

Hình 2-2-68 nêu ra lưu đồ của thuật toán để tính diện tích bằng việc dùng máy tính và phương pháp Simpson.

Hình 2-2-68 Lưu đồ của việc tính xấp xỉ diện tích bằng phương pháp Simpson



2.2.9 Thuật toán đối sánh

Trong thuật toán đối sánh, các giá trị được lưu giữ trong mảng được so sánh để thu được giải pháp. Việc duyệt các xâu kí tự được mô tả trong Mục 2.2.4, Xử lí xâu kí tự, cũng dùng một trong trong các thuật toán đối sánh.
Mục này mô tả cho bài toán hôn nhân ổn định để giải thích cho một trong những thuật toán đối sánh tiêu biểu.

(1) Bài toán hôn nhân ổn định

Trong việc giải bài toán hôn nhân ổn định, các cặp đàn ông và đàn bà ổn định được xác định. Ổn định có nghĩa là trạng thái trong đó đàn ông và đàn bà cảm thấy yêu mến nhiều với bạn tình của mình hơn là với người khác. Đặc biệt, giả sử rằng có ba đàn ông và đàn bà, đàn ông tên là A, B và C và đàn bà tên là a, b và c. Các mức yêu mến (từ cao tới thấp theo thứ tự 1, 2 và 3) được vẽ trong bảng sau:

<Đàn ông>

	1	2	3
A	B	a	c
B	C	A	b
C	C	B	a

<Đàn bà>

	1	2	3
a	A	C	B
b	C	B	A
c	A	B	C

Nếu họ được sắp thành cặp là
P = { (A, a), (B, b), (C, c) },
A cảm thấy yêu mến b hơn a, người là bạn tình hiện tại. b cảm thấy yêu mến B, người là bạn tình hiện tại, hơn A. Do đó, phải không có vấn đề gì. B cảm thấy yêu mến a và c hơn b, người là bạn tình hiện tại. Bởi vì a cảm thấy yêu mến A, người là bạn tình hiện tại, hơn B, nên phải không có vấn đề gì. Tuy nhiên, c cảm thấy yêu mến B hơn C người là bạn tình hiện tại. Trạng thái này được gọi là trạng thái không ổn định. Bằng việc thay đổi bạn tình, họ có thể được sắp thành cặp là
P = { (A, a), (B, c), (C, b) }.
Trạng thái sắp cặp này có thể được phân tích như sau:

① A cảm thấy yêu mến b hơn a, người là bạn tình hiện tại. → b cảm thấy yêu mến C, người là bạn tình hiện tại, hơn A.
② B cảm thấy yêu mến c người là bạn tình hiện tại.
③ C cảm thấy yêu mến c hơn b, người là bạn tình hiện tại. → c cảm thấy yêu mến B, người là bạn tình hiện tại, hơn C.
④ a cảm thấy yêu mến A người là bạn tình hiện tại.
⑤ b cảm thấy yêu mến C, người là bạn tình hiện tại.
⑥ c cảm thấy yêu mến A hơn B, người là bạn tình hiện tại. → A cảm thấy yêu mến a, người là bạn tình hiện tại, hơn c.

Trong việc sắp xếp cặp này, không cặp nào có nhân tố không ổn định. Kết quả này được gọi là ổn định hay đối sánh ổn định.
Đối sánh ổn định có thể không nhất thiết được xác định duy nhất. Có thể có trường hợp việc đối sánh ổn định không thể được đạt tới bằng việc dùng cách tiếp cận được mô tả ở trên. Bài toán hôn nhân ổn định được mô tả tại đây, được thiết kế để đạt tới việc sánh đôi ổn định bằng cách xác định các điều kiện cho từng cặp nhất định:

[Bài toàn hôn nhân ổn định]

N đàn ông và N đàn bà đang tìm các cặp ổn định.

- Các mức yêu mến của đàn ông và đàn bà được đặt sẵn trong mảng M và F (số các phần tử : N+1).
- Xem như các mức yêu mến, các số từ 1 tới 5 (cao tới thấp) được gán cho các số của từng cặp.

Ví dụ Nếu có năm đàn ông và năm đàn bà

[Tổ hợp M: Các mức yêu mến được xét từ phía đàn ông]

	1	2	3	4	5	6 (PT)
M (1)	2	1	4	3	5	0
M (2)	1	3	4	2	5	0
M (3)	1	4	5	2	3	0
M (4)	5	4	1	3	2	0
M (5)	4	2	3	1	5	0

* Một bạn tình được đưa vào trong PT.
0 được đặt ở đây như giá trị ban đầu.

[Tổ hợp F: Các mức yêu mến được xét từ phía đàn bà]

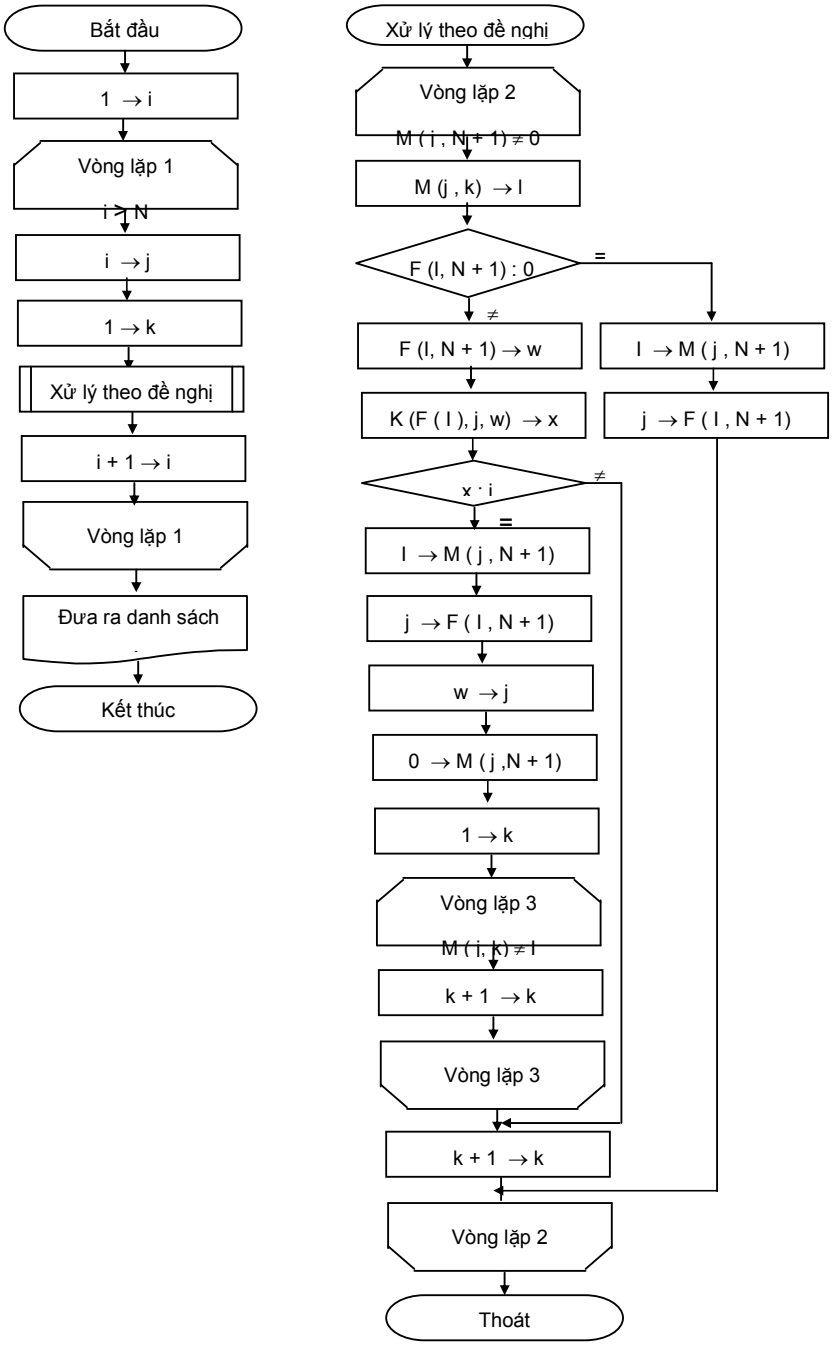
	1	2	3	4	5	6 (PT)
F (1)	3	1	5	2	4	0
F (2)	2	1	4	3	5	0
F (3)	3	2	5	4	1	0
F (4)	5	2	1	3	4	0
F (5)	5	4	2	1	3	0

* Một bạn tình được đưa vào trong PT.
0 được đặt ở đây như giá trị ban đầu.

- Các đề nghị được thực hiện theo thứ tự của M (1), M (2).... M (N).
- Đàn ông đưa ra đề nghị với đàn bà theo thứ tự cao xuống thấp của yêu mến. Họ lặp lại các đề nghị cho tới khi họ thu được OK. Tuy nhiên có thể xảy ra là các OK thu được như vậy có thể bị xoá bỏ.
- Đàn bà cho OK hoặc NO theo tiêu chuẩn sau:
  ① Nếu một người nhận đề nghị lần đầu tiên, cô ấy cho OK và bảo đảm anh ta như bạn tình.
  ② Nếu cô ấy đã có một đàn ông, cô ấy so sánh độ yêu mến của mình với bạn tình hiện tại và độ yêu mến với người đã làm đề nghị.
- Nếu cô ấy cảm thấy yêu mến bạn tình hiện tại hơn, cô ấy cho NO.
- Nếu cô ấy cảm thấy yêu mến người đàn ông đưa ra đề nghị, cô ấy xoá bỏ bạn tình hiện tại và bảo đảm với người vừa đưa ra đề nghị làm bạn tình (OK).
- Người đàn ông vừa được OK nhưng lại bị xoá bỏ về sau bắt đầu làm đề nghị với những đàn bà ở cấp yêu mến cao hơn. Anh ta lặp lại việc làm đề nghị cho tới khi anh ta thu được OK.

Trong lưu đồ được vẽ trong Hình 2-2-69, hàm K (p1, p2, p3) được dùng để so sánh các mức yêu mến.
Hàm K (p1, p2, p3): Hàm này là để cho lại hoặc p2 hoặc p3 trong phần tử mảng p1, phần tử cao hơn theo mức độ yêu mến. K (F(2), 1, 3) $\rightarrow$ 1
Hình 2-2-69 nêu lưu đồ của thuật toán của bài toán hôn nhân ổn định.

Hình 2-2-69 Lưu đồ của thuật toán của bài toán hôn nhân ổn định



2.2.10 Thuật toán xấp xỉ và xác suất

Các thuật toán nói chung được dùng để thu lấy các giá trị đúng hay giải pháp. Có các bài toán đòi hỏi thời gian rất dài để giải, và có những bài toán hiện không có thuật toán giải. Trong trường hợp này, các thuật toán để thu được lời giải có sai số hay có xác suất chứa sai số rất nhỏ sẽ được dùng.

(1) Thuật toán xấp xỉ

Thuật toán xấp xỉ được dùng để thu lấy lời giải xấp xỉ trong những trường hợp không thể thu được lời giải đúng cho bài toán hay phải mất rất nhiều thời gian giải. Cả hai phương pháp Newton và Simpson đã mô tả trong Mục 2. 2. 8 rơi vào loại thuật toán xấp xỉ.
Mục này mô tả cho bài toán ba lô xem như một thuật toán xấp xỉ tiêu biểu.

[Bài toán ba lô]

Bạn có n món hàng khác nhau về trọng lượng và giá trị. Bạn muốn bán chúng trong thị trấn nhưng bạn không thể để tất cả chúng vào ba lô của mình được. Hãy tìm ra cách tổ hợp hàng hoá làm tối đa giá trị của chúng.

Chúng ta áp dụng các giá trị sau để giải thích cách thuật toán làm việc:
Số hàng hoá (khoản mục): 5
Trọng lượng từng khoản mục (kg): {2, 3, 5, 7, 8} áp dụng cho khoản mục 1, khoản mục 2,.....khoản mục 5 theo thứ tự đó
Giá trị của từng khoản mục (10,000 yen):{2, 4, 8, 9, 10} áp dụng cho khoản mục 1, khoản mục 2, …khoản mục 5 theo thứ tự đó.
Khối lượng của ba lô: 10 kg
Một cách để giải bài toán này là xem xét từng tổ hợp hàng hoá mà sẽ đem trọng lượng tới 10 kg và so sánh giá trị toàn bộ của hàng hoá trong từng tổ hợp. Với bài toán đặc biệt này, tổng giá trị trở thành cao nhất nếu hàng hoá 1, 2 và 3 được đóng gói vào ba lô, như được nêu trong bảng dưới đây:

Hàng được chọn	Tổng trọng lượng	Tổng giá trị	
Hàng 1, 2 và 3	2 + 3 + 5 = 10	2 + 4 + 8 = 14	← Tối đa
Hàng 1 và 5	2 + 8 = 10	2 + 10 = 12	
Hàng 2 và 4	3 + 7 = 10	4 + 9 = 13	

Việc làm tổng trọng lượng bằng khối lượng của ba lô không phải bao giờ cũng là giải pháp tốt nhất. Chẳng hạn, nếu có khoản mục thứ sáu, nặng 9 kg và có giá trị 150,000 yên thì giá trị tối đa có thể thu được bởi việc đóng gói mỗi một khoản mục này vào ba lô. Để tìm ra giải pháp tốt nhất, tất cả các tổ hợp hàng hoá có thể đều phải được xem xét. Khi hàng hoá tăng lên về số lượng, thì số tổ hợp trở thành khổng lồ và phải mất nhiều thời gian để tìm ra giải pháp. Xem như giải pháp cho điều này, có thể dùng thuật toán xấp xỉ.
Bài toán ba lô có thể được phát biểu như sau:

Có n số nguyên dương a_1, a_2, a_3, ..., a_n, và b_1, b_2, b_3, ..., b_n, và một số nguyên dương, c. Hãy tìm một tổ hợp của x_1, x_2, x_3, ..., x_n làm tối đa tổng số của b_ix_i (i = 1-n) với x_i ={0, 1} và tổng số của a_ix_i (i = 1-n) là bằng hay nhỏ hơn c.

Nếu phát biểu này được xét tương tự với bài toán đã được mô tả trước đây, thì a là trọng lượng của hàng hoá, b là giá trị của hàng hoá, và c là dung lượng của ba lô. x là liệu hàng hoá có được đóng gói vào ba lô hay không. 0 nghĩa là hàng hoá không được đóng gói còn 1 nghĩa là hàng hoá được đóng gói. Do đó, bài toán ba lô và lời giải có thể được diễn đạt như sau:

a = | 2, 3, 5, 7, 8 |
b = | 2, 4, 8, 9, 10 |
c = 10
x = | 1, 1, 1, 0, 0 |

Bằng việc dùng công thức này, việc duyệt được thực hiện 2^n lần để kiểm tra mọi tổ hợp có thể của 0, 1 trong mảng x nếu thuật toán xấp xỉ không được dùng.

Tuy nhiên, dùng thuật toán xấp xỉ, các giá trị đơn vị của mọi hàng hoá được nhận diện trước hết. Giá trị đơn vị nghĩa là giá trị trên trọng lượng và nó được cho bởi giá trị (b_i)÷trọng lượng (a_i).

Giá trị đơn vị k = | 2/2, 4/3, 8/5, 9/7, 10/8 |
= | 1.00, 1.33, 1.60, 1.28, 1.25 |

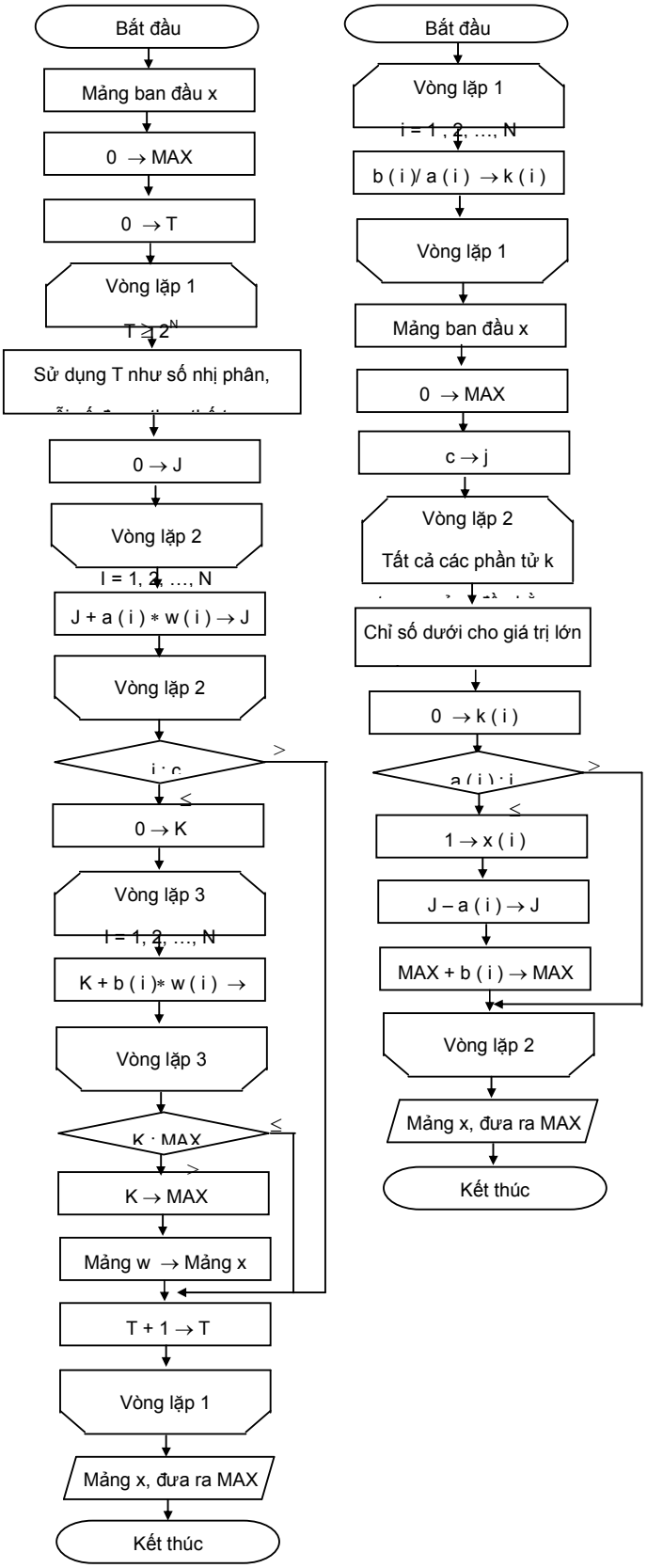
Sau khi tất cả các giá trị đơn vị đã được nhận diện, hàng hoá được đóng gói vào ba lô theo thứ tự giá trị đơn vị cao tới thấp. Bởi vì hàng hoá không thể được phân chia, nên hàng hoá vượt quá khối lượng không dùng của ba lô không thể được đóng gói.

① Khoản mục 3 với giá trị đơn vị cao nhất được đóng gói: khối lượng không dùng = 10-5 = 5
② Khoản mục 2 với giá trị đơn vị cao thứ hai được đóng gói: khối lượng không dùng = 5-3 = 2
③ Khoản mục 4 với giá trị đơn vị bé hơn khoản mục 2 không thể được đóng gói: khối lượng không dùng < trọng lượng của khoản mục 4
④ Khoản mục 5 với giá trị đơn vị bé hơn khoản mục 4 không thể được đóng gói: khối lượng không dùng < trọng lượng khoản mục 5
⑤ Khoản mục 1 với giá trị đơn vị bé hơn giá trị của khoản mục 5 được đóng gói: khối lượng không dùng = 2-2 = 0

Theo cách này để thu được giải pháp không nhất thiết cho lời giải tốt nhất. Nếu các giá trị được xác định là {2, 4, 8, 11, 10}, thì thuật toán xấp xỉ cho lời giải : (khoản mục 1, khoản mục 2, khoản mục 3) = 140,000 yen mặc dầu chúng ta đã có lời giải tốt nhất: (khoản mục 2, khoản mục 4) = 150,000 yen. Tuy nhiên lời giải xấp xỉ được dùng trong nhiều lĩnh vực khác nhau như phương pháp thu được nhanh chóng giá trị xấp xỉ rất gần với lời giải tốt nhất.

Hình 2-2-70 đưa ra lưu đồ của thuật toán để giải bài toán ba lô.

Hình 2-2-70 Lưu đồ của thuật toán để giải bài toán ba lô



(2) Thuật toán xác suất

Thuật toán xác suất dùng cách tiếp cận xác suất để tìm ra lời giải bằng số ngẫu nhiên. Cách tiếp cận xác suất là cách tiếp cận trong đó sự thích hợp của lời giải được xem xét dưới dạng xác suất, hoặc là cách tiếp cận mà giải pháp được tìm ra dựa trên xác suất xuất hiện của biến cố nào đó.
Mục này mô tả về bài toán kiểm thử số nguyên tố như thuật toán để xem xét tính thích hợp của lời giải dưới dạng xác suất, và trường hợp thu được hằng số hình tròn π như thuật toán để tìm lời giải dựa trên xác suất xuất hiện của biến cố nào đó.

① Bài toán kiểm thử số nguyên tố

Bài toán kiểm thử số nguyên tố là một tiến trình kiểm tra số đã cho N (= 2^Sd+1, d là số lẻ) và xác định xem nó có phải là số nguyên tố hay không. Để giải bài toán kiểm thử số nguyên tố, số N đã cho được chia ra cho các số nguyên 2, 3, ..., $\sqrt{N}$, và nó được coi là số nguyên tố nếu nó không chia hết cho bất kì số nguyên nào. Mặc dầu lời giải đúng có thể thu được bằng việc dùng cách tiếp cận này, cần thời gian dài hơn nhiều để đạt tới lời giải nếu giá trị của N là lớn. Xem như một lời giải ta sử dụng thuật toán Rabin, được thiết kế với định lí sau:

[Định lí số nguyên tố]

Nếu N (= 2^Sd+1, d là số lẻ) là số nguyên tố, thì một trong hai điều kiện được nêu dưới đây là đúng cho bất kì số nguyên dương được chọn tuỳ ý, a (>1):
- $a^d = 1 \pmod N$
- $a^{2kd} = -1 \pmod N$ với $0 \le k \le S-1$

Nếu số nguyên được chọn tuỳ ý, a, không đáp ứng các điều kiện trên, thì N được xem như hợp số, không phải là số nguyên tố. Xác suất số nguyên được chọn tuỳ ý, a, đối với hợp số N có thể đáp ứng các điều kiện trên là bằng hay bé hơn 1/4. Do đó, nếu số nguyên được chọn tuỳ ý, a, đáp ứng các điều kiện trên, thì xác suất xuất hiện của đánh giá rằng N là số nguyên tố là bằng hay lớn hơn 3/4.

	Xác suất các điều kiện có thể được đáp ứng	Xác suất các điều kiện không thể được đáp ứng
Số nguyên tố N	100%	0%
Hợp số N	25% hay thấp hơn	75% hay cao hơn

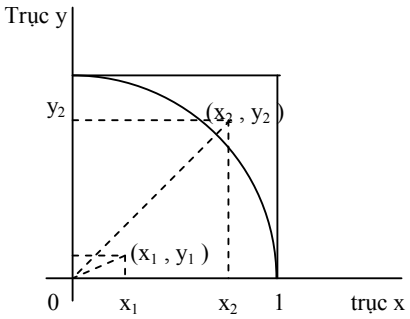
Nếu thuật toán này được dùng, thì khả năng còn lại là lời giải được cho bởi thuật toán này là không đúng. Kiểu thuật toán này được gọi là thuật toán xác suất với sai số bị chặn. Với thí dụ đặc biệt trên, vì xác suất mà sai số sẽ xuất hiện là đủ thấp, nên lời giải được cho bởi thuật toán này nên được xem là phụ thuộc. Bên cạnh thuật toán xác suất với sai số bị chặn, thuật toán xác suất không sai số đôi khi cũng được dùng. Mặc dầu thuật toán này về mặt lí thuyết có thể đảm bảo tính đúng đắn của lời giải đã cho, đôi khi phải mất rất lâu mới thực hiện xong tiến trình thuật toán này, hay nó kết thúc mà không cho lời giải xác định.
Thuật toán xác suất không sai số là thuật toán sắp xếp nhanh. Thuật toán này được thiết kế để tăng cường hiệu quả bằng việc bố trí lại ngẫu nhiên dữ liệu vào bằng việc dùng số ngẫu nhiên.

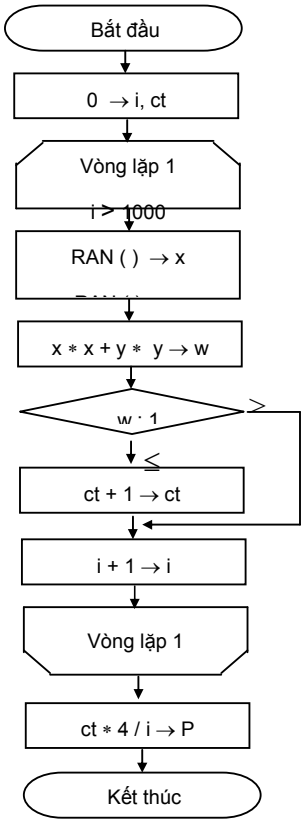
② Cách tìm hằng số hình tròn π

Hình 2-2-71 đưa ra một vòng tròn bán kính 1. Diện tích được bao bởi các trục và cung là một phần tư diện tích của hình tròn đầy đủ có bán kính 1. Do đó, nó là π/4 (= 1 × 1 × π/4).

Mặt khác, diện tích của hình vuông được bao bởi bốn đường, x = 0, x = 1, y = 0 và y = 1 là 1. Nếu các điểm trên hình vuông này được chọn lựa theo cách ngẫu nhiên, thì xác suất các điểm được chọn này ở bên trong vòng tròn là π/4.
Trong lưu đồ được vẽ trong Hình 2-2-71, 1,000 điểm được sinh ra bằng việc dùng hàm RAN ( ) mà có thể sinh số ngẫu nhiên giữa 0 và 1. Liệu các điểm được sinh ra có ở bên trong vòng tròn hay không được xác định bằng việc đo khoảng cách trực tiếp từ (0, 0) tới từng điểm. Để làm đơn giản hoá tính toán, việc tính căn √ bị bỏ qua. Trong trường hợp của ví dụ được nêu trong Hình 2-2-71, người ta đánh giá rằng (x1, y1) là bên trong nếu $(x1^2+y1^2)\leq 1$ và rằng (x2, y2) là ở bên ngoài nếu $(x2^2+y2^2)>1$.

Hình 2-2-71 Lưu đồ của thuật toán tìm hằng vòng tròn π





Vì hằng vòng tròn thu được theo cách này chứa sai số nảy sinh từ các đặc trưng cố hữu trong phương pháp được dùng để sinh ra số ngẫu nhiên, nó nói chung được dùng trong biểu diễn có chứa sai số chuẩn (lời giải ± sai số chuẩn).
Thuật toán giống thế này, dùng số ngẫu nhiên để giải các bài toán toán học, được gọi là phương pháp Monte Carlo.

2.3 Đánh giá thuật toán

Thuật toán nên được đánh giá và lựa chọn dựa trên những tiêu chí nào đó. Nếu có thể tìm được thuật toán thích hợp nhất để giải một bài toán, thì công việc lập trình có thể được thực hiện một cách có hiệu quả và chương trình chất lượng cao có thể được tạo ra.
Mục này mô tả cho ba tiêu chí đánh giá thuật toán sau:

- Đánh giá theo độ phức tạp tính toán (tiêu chí đánh giá tính hiệu quả)
- Đánh giá theo tính hợp lệ (tiêu chí để đánh giá độ tin cậy)
- Đánh giá theo việc biểu diễn (tiêu chí để đánh giá việc xoá bỏ dư thừa và làm tăng tốc độ xử lí)

2.3.1 Đánh giá theo độ phức tạp tính toán

Độ phức tạp tính toán, cũng được gọi là độ đo tính toán, chỉ ra về mặt định lượng tải công việc cần thực hiện tính toán.
Độ phức tạp tính toán là độ đo được dùng để làm rõ ràng về mặt toán học máy tính yêu cầu bao nhiêu thời gian và vùng bộ nhớ để thực hiện tính toán.
Độ phức tạp tính toán được biểu diễn bởi O (cấp).
Ví dụ: Nếu một thuật toán giải quyết các dữ liệu theo cách dữ liệu tăng lên gấp đôi, gấp ba, gấp tư....v.v., thời gian thực hiện cũng trở nên gấp đôi, gấp ba, gấp tư...v.v., thì độ phức tạp tính toán của thuật toán này là $O(n)$.

(1) Các kiểu độ phức tạp tính toán

Có hai kiểu độ phức tạp tính toán.

- Độ đo phức tạp thời gian: Thời gian tối đa thuật toán cần để xử lí mọi dữ liệu
- Độ phức tạp không gian: Vùng tối đa thuật toán cần để xử lí mọi dữ liệu

Nói chung, độ phức tạp tính toán giả thiết trường hợp tồi nhất. Độ phức tạp tính toán được thảo luận trong mục này cũng giả thiết trường hợp tồi nhất, nếu không được nói thêm. Nếu độ phức tạp tính toán trung bình là $O(n\log n)$ và nếu độ phức tạp tính toán tối đa là $O(n^2)$, như trường hợp của sắp xếp nhanh, thì độ phức tạp tính toán trung bình nên được coi là quan trọng nhất.
Độ phức tạp tính toán của một thuật toán được biểu diễn như kích cỡ dữ liệu (chẳng hạn, hình vuông với cạnh n) là không bị hạn chế bởi những giới hạn phần cứng hay phần mềm.

(2) Ví dụ tính toán

Bây giờ chúng ta lấy thuật toán duyệt tuyến tính (duyệt tuần tự) và thuật toán duyệt nhị phân làm ví dụ. Giả sử rằng kích cỡ dữ liệu là n, số lần so sánh và độ phức tạp tính toán là như sau:

① Duyệt tuyến tính (duyệt tuần tự)

- Số tối thiểu lần so sánh: 1 (độ phức tạp tính toán: $O(1)$)
- Số trung bình lần so sánh: $n/2$ (độ phức tạp tính toán: $O(n)$)
- Số tối đa lần so sánh: n (độ phức tạp tính toán: $O(n)$)

② Duyệt nhị phân

- Số tối thiểu lần so sánh: 1 (độ phức tạp tính toán: $O(1)$)
- Số trung bình lần so sánh: $[\log_2 n]$ (độ phức tạp tính toán: $O(\log_2 n)$)
- Số tối đa lần so sánh: $[\log_2 n]+1$ (độ phức tạp tính toán: $O(\log_2 n)$)

2.3.2 Đánh giá theo tính hợp lệ

Tính hợp lệ của thuật toán là tiêu chí để đưa ra đánh giá xem liệu một thuật toán có thoả mãn đặc tả chương trình hay không (tài liệu thiết kế mô đun v.v.).
Tính hợp lệ có thể được chia thêm thành ba loại:

- Hợp lệ bộ phận: Liệu các đoạn hay hàm có thoả mãn đặc tả hay không.
- Tính kết thúc: Liệu một chương trình có thể kết thúc sau một số xác định các bước thực hiện hay không, như đã được định nghĩa bởi thuật toán; bởi vì chu trình vô hạn phải không được xuất hiện.
- Tính hợp lệ toàn bộ: Liệu toàn thể thuật toán có thoả mãn đặc tả hay không.

Mặc dầu các phương pháp khác nhau được dùng để kiểm chứng tính hợp lệ của thuật toán, vẫn khó kiểm chứng nó một cách đầy đủ. Nếu phạm vi dữ liệu được xác định, thì tính hợp lệ có thể được kiểm chứng dễ dàng bằng việc phát hiện rắc rối có thể xuất hiện khi dữ liệu bên ngoài phạm vi đã xác định được đưa vào.

2.3.3 Đánh giá theo biểu diễn

Các thuật toán phải được đánh giá theo không chỉ độ phức tạp tính toán và tính hợp lệ, mà cũng còn theo cách biểu diễn thuật toán, thường được gọi là khởi thảo biểu diễn thuật toán.

Ví dụ

- Cùng một bước được thực hiện lặp lại: nếu các bước được lặp lại được xác định như các trình con, thì chúng có thể được mô tả như một bước sao cho luồng các bước trong thuật toán có thể được trình bày theo cách đơn giản, dễ hiểu.
- Cần tăng tốc độ thực hiện thuật toán: Cần phải chú ý tới các bước được lặp lại thường xuyên nhất, và cách làm tăng tốc độ nên được nghiên cứu (nếu sắp xếp nhanh được dùng, có thể cần bỏ việc dùng lời gọi đệ qui và làm việc với phương án khác).

2.4 Cách thiết kế thuật toán

Dựa trên dữ liệu thu được bởi việc phân tích bài toán và những kết quả đánh giá được mô tả ở mục trước, thuật toán được thiết kế với sự chú ý nhiều nhất tới cách lời giải có thể có được với mức độ hiệu quả cao. Có nhiều phương pháp được dùng để thiết kế thuật toán. Các phương pháp tiêu biểu là:

- Phương pháp qui hoạch động
- Phương pháp thuật toán tham lam
- Phương pháp rút gọn

(1) Phương pháp qui hoạch động

Trong phương pháp qui hoạch động, một bài toán được xét là một tập các bài toán con. Các bước cần tuân theo là:

1. Bài toán được chia thành một số bài toán con.
2. Giải pháp được tìm cho từng bài toán con.
3. Một số bài toán con được gắn lại để làm ra bài toán bộ phận hơi lớn hơn.
4. Các bước 2 và 3 được lặp lại cho tới khi phương pháp giải pháp cho bài toán gốc được tìm ra.

(2) Phương pháp thuật toán tham lam

Trong phương pháp thuật toán tham lam, được dùng để tìm lời giải cho bài toán tối ưu, các giá trị của các biến được thay đổi theo mức độ phụ thuộc vào từng điều kiện hiện tại. Chẳng hạn, để tìm cách làm giảm tối thiểu số đồng xu phải trả cho một khoản tiền xác định, các số đồng xu mệnh giá lớn tới nhỏ được xác định bằng việc dùng phương pháp thuật toán tham lam.

Ví dụ

Để trả 574 yen
Một xu 500-yen - con dư 74 yen
Một xu 50-yen - còn dư 24 yen
Hai xu mười yen - còn dư 4
Bốn xu 1-yen - Không còn dư

(3) Phương pháp rút gọn

Trong phương pháp rút gọn, thuật toán gốc được thiết kế có độ phức tạp $O(n)$ được giả thiết là một tiến trình rút gọn chiều dài thời gian cn, sao cho nó thể được rút gọn thành kích cỡ nhỏ hơn để tạo ra thuật toán mới cấp $O(n/x)$. Nếu $O(n)>cn+O(n/x)$ không thể được thoả mãn, thì thuật toán này là vô nghĩa. Do đó, nếu n là rất nhỏ, thuật toán này không thể tạo ra kết quả mong muốn được.

Bài tập

Q1 **Thuật ngữ nào được dùng để chỉ ra một thuật toán duyệt các phần tử một cách tuần tự từ đầu tới cuối một bảng?**

a. Tuyến tính b. Nhị phân c. Băm d. Đống

Q2 **Giá trị trong mảng có chứa n phần tử, được so sánh tuần tự với dữ liệu X là dữ liệu được duyệt. Nếu dữ liệu X sánh với giá trị nào đó trong bảng, "exist" được chỉ ra. Dữ liệu X được lưu giữ vào vị trí được lấy chỉ số là n+1.**

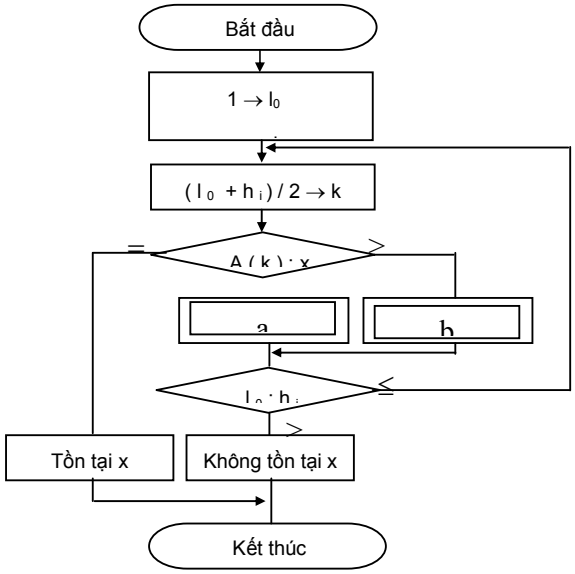
Chỉ số	1	2	3	...	i	...	n	n+1
Giá trị	a_1	a_2	a_3	...	a_i	...	a_n	X

Trong thuật toán duyệt tuyến tính được nêu dưới đây, điều kiện nào nên được đưa vào trong chỗ trống ☐ ?

Bước 1 1 được đặt cho chỉ số i.
Bước 2 nếu ☐, trình nhảy tới bước 5.
Bước 3 1 được cộng vào chỉ số i.
Bước 4 Trình nhảy tới bước 2.
Bước 5 Nếu chỉ số i là n hay bé hơn, "exist" được chỉ ra.
Bước 6 Kết thúc

a. $i \geq n$ b. $i \neq n$ b. $i < n$ d. $X = a_i$ e. $X \neq a_i$

Q3 **Có mảng A chứa n dữ liệu được sắp theo thứ tự tăng dần. Lưu đồ sau đây nêu ra một trình để tìm kiếm dữ liệu x từ mảng A bằng việc dùng phương pháp duyệt nhị phân. Hãy chọn một tổ hợp đúng các thao tác và đưa chúng vào chỗ trống a và b. Các số thập phân của một giá trị thu được bằng việc chia phải bị chặt đi.**



	a	b
a	k + 1 → hi	K - 1 → lo
b	k - 1 → hi	k + 1 → lo
c	k + 1 → lo	K - 1 → hi
d	k - 1 → lo	k + 1 → hi

Q4 **Có một bảng có 2,000 phần tử khác nhau được sắp theo thứ tự tăng dần của khoá. Dùng một khoá đưa vào từ ngoài, bảng này được duyệt theo phương pháp duyệt nhị phân, và các phần tử sánh đúng với khoá được tìm ra. Số tối đa lần so sánh cần thực hiện trước khi tất cả mọi phần tử sánh đúng được tìm thấy là gì? Giả sử rằng bảng này chứa các khoá sánh đúng.**

a. 10 b. 11 c. 12 d. 13

Q5 **Chú thích nào trong các chú thích sau được nêu về phương pháp duyệt là <u>sai</u>?**

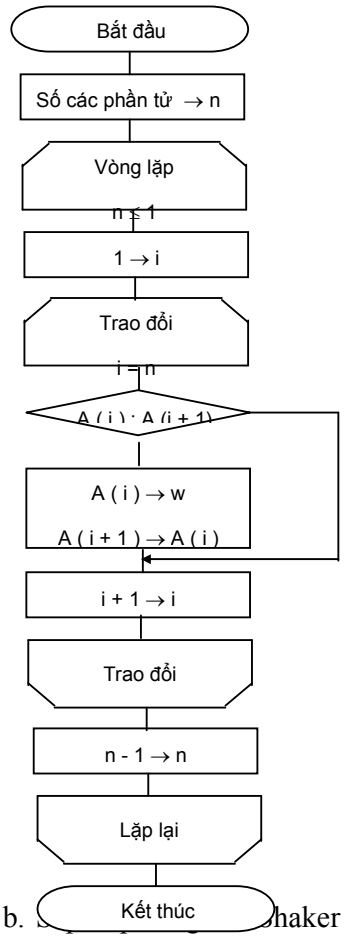
a. Dùng phương pháp duyệt nhị phân, dữ liệu phải được sắp xếp.
b. Để duyệt 100 dữ liệu bằng việc dùng phương pháp duyệt nhị phân, số lần so sánh tối đa được cần tới để tìm ra dữ liệu đích là 7.
c. Nếu phương pháp duyệt tuyến tính được dùng, số lần so sánh không nhất thiết giảm đi cho dù dữ liệu đã được lưu giữ.
d. Nếu số dữ liệu là 10 hay nhỏ hơn, thì số lần so sánh trung bình mà phương pháp duyệt tuyến tính đòi hỏi, là nhỏ hơn số lần trung bình của phương pháp duyệt nhị phân.
e. Nếu số dữ liệu tăng lên từ 100 tới 1,000, thì số lần so sánh tăng lên 10 lần hơn phương pháp duyệt tuyến tính được dùng. Bằng việc dùng phương pháp duyệt nhị phân, số này tăng lên hai hay ít hơn.

Q6 **Liên quan tới sắp xếp và gộp dữ liệu, hãy chọn những tổ hợp đúng của các từ và đưa chúng vào trong chỗ trống [ ].**

Việc sắp xếp dữ liệu theo thứ tự giá trị nhỏ tới lớn được tham chiếu là [A] [B]. Nếu một dãy dữ liệu đích ở trong bộ nhớ phụ, phép toán này được gọi là [C].
Việc tích hợp hai hay nhiều tệp [D] theo thứ tự nào đó vào một tệp được gọi là [E].

	A	B	C	D	E
a	giảm dần	sắp xếp	sắp xếp ngoài	được sắp xếp	việc gộp
b	tăng dần	gộp	gộp ngoài	được gộp	việc sắp xếp
c	giảm dần	gộp	gộp trong	được gộp	việc sắp xếp
d	tăng dần	sắp xếp	việc sắp xếp ngoài	được sắp xếp	việc gộp
e	tăng dần	gộp	gộp trong	được gộp	việc sắp xếp

Q7 Lưu đồ sau đây vẽ cho loại thuật toán nào?



a. Sắp xếp nhanh-Quick sort b. [illegible]haker sort c. Sắp xếp bóc vỏ-Shell sort

d. Sắp xếp chèn thêm-Insertion sort e. Sắp xếp nổi bọt-Bubble sort

Q8 Phải mất 1.0 giây cho máy tính nào đó sắp xếp 1,000 dữ liệu bằng việc dùng phương pháp sắp xếp nổi bọt. Phải mất thời gian bao lâu để sắp 100,000 dữ liệu cùng kiểu? Thời gian mất cho một máy tính sắp xếp dữ liệu kiểu nổi bọt là tỉ lệ với bình phương của số dữ liệu được xác định là n.

a. 1 b. 10 c. 100 d. 1,000 e. 10,000

Q9 Liên quan tới phương pháp sắp xếp dữ liệu, mô tả nào trong các mô tả được cho dưới đây là đúng?

a. Sắp xếp nhanh Quick sort là phương pháp sẵp xếp dữ liệu theo các dãy con bao gồm các khoản mục dữ liệu được lấy từ các khoảng và sắp xếp các dãy con nhỏ hơn bao gồm các

khoản mục dữ liệu được lấy tại khoảng nhỏ nhất.

b. Sắp xếp bóc vỏ Shell sort là phương pháp sắp xếp dữ liệu bằng cách so sánh một cặp các phần tử kề và tráo đổi chúng nếu phần tử thứ hai lớn hơn phần tử thứ nhất.

c. Sắp xếp nổi bọt Bubble là phương pháp sắp xếp dữ liệu bằng cách đặt một giá trị tham chiếu trung gian, phân bổ các phân tử với giá trị lớn hơn giá trị tham chiếu trong phần này và đặt các phần tử với gí trị nhỏ hơn giá trị tham chiếu vào phần kia và lặp lại việc này cho từng phần riêng một.

d. Sắp xếp vun đống là phương pháp sắp xếp dữ liệu bằng việc biểu diễn một vùng chưa sắp xếp như một cây con, lấy giá trị tối đa hay tối thiểu từ miền chưa sắp, chuyển giá trị tối đa hay tối thiểu vào vùng được sắp xếp và lặp lại việc này để làm hẹp dần miền chưa sắp xếp.

Q10 Mô tả nào là mô tả thích hợp cho sắp xếp nhanh quick sort?

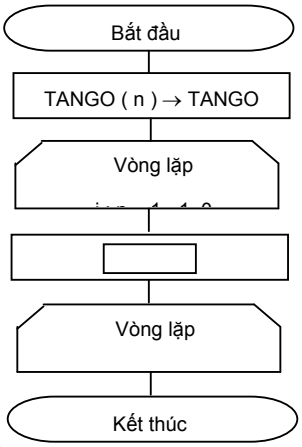
a. So sánh và tráo đổi được thực hiện cho hai dữ liệu xa nhau với khoảng cách nào đó. Khoảng cách này dần dần và liên tục được làm hẹp để sắp xếp mọi dữ liệu.

b. Giá trị tối thiểu thứ nhất được tìm ra trong dữ liệu. Giá trị tối thiểu thứ hai được tìm ra trong dữ liệu mà trong đó giá trị tối thiểu thứ nhất không được bao hàm. Việc này được thực hiện lặp lại.

c. Dữ liệu được chia thành một nhóm các dữ liệu nhỏ hơn một giá trị tham chiếu và nhóm kia là các dữ liệu lớn hơn giá trị tham chiếu. Trong từng nhóm, một giá trị tham chiếu mới được lựa ra và dữ liệu giống thế lại được phân chia thành hai nhóm dựa trên giá trị tham chiếu này. Việc này được thực hiện lặp lại.

d. Dữ liệu kề được so sánh và tráo đổi lặp lại để cho phép dữ liệu nhỏ hơn được chuyển về cuối mảng dữ liệu.

Q11 Có mảng TANGO với số chỉ số bắt đầu từ 0. n từ được chứa trong TANGO (1) tới TANGO(n). Lưu đồ dưới đây nêu ra các bước để tổ chức lại bảng từ bằng việc dịch chuyển các từ trong TANGO (1) tới TANGO (n-1) ngược lại, một từ, để đặt từ thứ n vào TANGO (1). Hãy đưa vào một bước đúng trong chỗ trống ☐.



a. TANGO (i) → TANGO (i+1)

b. TANGO (i) → TANGO (n-i)

d. TANGO (i+1) → TANGO (n-i)

e. TANGO (n-i) → TANGO (i)

Q12 **Dựa trên mô tả được nêu cho phương pháp Newton, được biết tới như thuật toán để thu được lời giải xấp xỉ cho phương trình f(x) = 0, hãy chọn mô tả nào thích hợp nhất sau đây.**

a. Mặc dầu một hàm, f(x), không thể được lấy vi phân, có thể thu được một lời giải xấp xỉ.
b. Như được thấy theo quan điểm hình học, phương pháp thu lấy lời giải xấp xỉ bằng việc dùng đường tiếp tuyến của y = f(x).
c. Hai giá trị khởi đầu khác nhau phải được cung cấp.
d. Với bất kì giá trị ban đầu nào bao giờ cũng thu được một giá trị xấp xỉ.

3 Thiết kế trong

Mục đích của chương

Thiết kế trong là một tiến trình thiết kế ra hệ thống dựa trên tài liệu thiết kế ngoài. Nó giải quyết cách máy tính cần phải vận hành để thực hiện nhiệm vụ đã trao, trong khi thiết kế ngoài tập trung vào chức năng, tính hiệu quả và dễ dùng theo quan điểm người sử dụng.
Chương này mô tả các nhiệm vụ thiết kế trong cần được thực hiện để thiết kế sản phẩm.

① Hiểu mục đích, các điểm quan trọng và các thủ tục của thiết kế trong.
② Hiểu nội dung và ý nghĩa của từng tiến trình thiết kế trong.
③ Hiểu việc tạo ra và dùng lại các bộ phận cũng như các phương tiện thực tế để vật chất hoá việc tạo ra và dùng lại các bộ phận

Giới thiệu

Mục đích của thiết kế trong là để xác định các chức năng của phần mềm theo quan điểm của người phát triển. Thiết kế trong là một tiến trình rất quan trọng, vì một kế hoạch thiết kế cơ sở được phát biểu ra và phân tích hệ thống phải được cài đặt trong tiến trình này.
Mặc dầu thiết kế ngoài không giải quyết với chương trình, thiết kế trong lại liên quan tới chương trình, mà trong đó các hàm cần thiết được xác định qua việc phát triển các hệ con trong giai đoạn thiết kế ngoài, phải được cài đặt. Chương trình có ảnh hưởng lớn, không chỉ lên thiết kế và kiểm thử chương trình trong tiến trình tiếp, mà còn lên tính hiệu quả khi hệ thống đi vào vận hành. Do đó, các hệ con phải được phân chia rất cẩn thận.
Trong chương này, chúng ta học cách từng khoản mục được tiến hành và kĩ thuật nào được dùng để cho ta có thể hiểu rõ một tài liệu thiết kế ngoài, và có khả năng tự bản thân mình chuẩn bị công việc thiết kế tin cậy.

3.1 Thiết kế trong là gì?

Thiết kế trong là thiết kế cho các phần không thấy được. Nó là thiết kế khi được nhìn theo quan điểm của chuyên gia máy tính hay người phát triển hệ thống. Trong thiết kế ngoài, hệ thống được thiết kế theo quan điểm của người dùng. Trong thiết kế trong, phần cứng và các hạn chế khác, cũng như các yêu cầu phần mềm được xem xét để làm cho máy tính có khả năng thực hiện chức năng được yêu cầu.

3.1.1 Mục đích của thiết kế trong và những điểm cần lưu ý

(1) Mục đích của thiết kế trong

Có hai mục đích của thiết kế trong:

- Xác định các chức năng mà phần mềm cần phải thực hiện theo quan điểm người phát triển.
  Tiến trình thiết kế trong trong tất cả các tiến trình phát triển hệ thống là nơi hệ thống được xác định theo quan điểm của người phát triển. Bởi vì các chức năng đã được xác định có ảnh hưởng lớn tới không chỉ thiết kế và kiểm thử chương trình trong các tiến trình sau, mà còn tới tính hiệu quả khi hệ thống vận hành, nên chúng phải được xác định rất cẩn thận.
- Đảm bảo sự độc lập của từng pha sao cho pha nọ có thể được phân biệt rõ ràng với pha kia.
  Mô hình thác đổ từ lâu đã được dùng như một phương tiện cho việc phát triển hệ thống. Đã từng được làm mịn và cải tiến, nó là phương pháp luận vẫn đang được sử dụng. Mặc dầu có thể xuất hiện một số chỗ chờm nhau giữa các pha cạnh nhau, từng pha về cơ bản vẫn phải độc lập.
  Cần chú ý tới chỗ công việc của thiết kế chương trình được thực hiện như một phần của

thiết kế trong.

Ví dụ Việc phân hoạch mô đun được coi là quan trọng hơn; cho nên nó được phân hoạch thành chương trình và từng chương trình lại được phân hoạch thêm nữa thành các mô đun.

Kết quả là chương trình bị lệch; nó hướng "phân hoạch mô đun" chứ không hướng "tiến trình", gây ra cho bản thân thiết kế trong mất tính mềm dẻo. Điều này có thể ảnh hưởng tới các giai đoạn kiểm thử, vận hành và thậm chí bảo trì.

(2) Các điểm cần lưu ý khi làm thiết kế trong

Khi làm thiết kế trong, bao giờ cũng phải luôn nhớ "tại sao", "cái gì" và "thế nào".

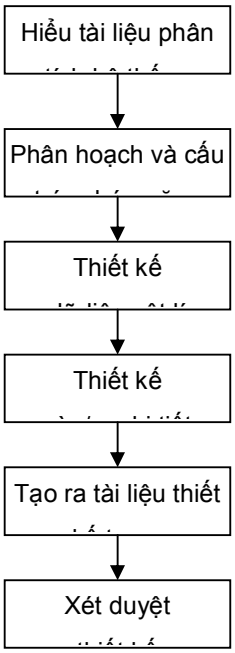
- Tại sao: Xác định các hệ con
  Tại sao từng hệ con được phân chia trong thiết kế ngoài lại phải được làm rõ ràng.
- Cái gì: Xác định các chức năng của hệ con
  Chức năng nào mà từng hệ con thực hiện trong luồng hệ thống tổng thể, phải được xem xét theo quan điểm về mối quan hệ giữa các hệ con.
- Thế nào: Xác định các chi tiết về hệ con trong chương trình
  Cách thức các điều kiện vào, xử lí chức năng và điều kiện ra cho việc vận hành của các hệ con có thể được cài đặt trong chương trình phải được làm sáng tỏ.

Trong khi làm sáng tỏ các nhân tố "tại sao" và "cái gì", xem như một nguyên tắc, thì "thế nào" phải bị gạt ra ngoài xem xét.

3.1.2 Thủ tục thiết kế trong

Hình 3-1-1 nêu ra thủ tục thiết kế trong.

Hình 3-1-1 Thủ tục thiết kế trong



Hình 1-1-1
Thủ tục
thiết kế trong

Chi tiết của từng bước được mô tả dưới đây:

(1) Hiểu tài liệu thiết kế ngoài

Trong giai đoạn thiết kế trong, nội dung của tài liệu thiết kế ngoài (sản phẩm hay tài liệu về tiến trình trước) phải được hiểu, tổ chức và phản ánh như một tổng thể trong hệ thống.
Tài liệu thiết kế ngoài chứa những thông tin sau:

- Tổng quan hệ thống (luồng hệ thống, luồng các hệ con v.v.)
  Thông tin này được dùng làm cơ sở cho việc phân chia một hệ con thành các phần chức năng.
- Thiết kế cái vào và cái ra.
  Thông tin này được dùng làm cơ sở cho việc chuẩn bị thiết kế vào ra chi tiết.
- Thiết kế dữ liệu logic.
  Thông tin này được dùng làm cơ sở cho việc chuẩn bị thiết kế dữ liệu vật lí.
- Cấu hình và hiệu năng hệ thống.
  Cấu hình hệ thống (phần cứng, phần mềm, mạng v.v..) và các giá trị ước lượng về hiệu năng hệ thống do thiết kế ngoài cung cấp, phải được làm hợp lệ khi chương trình được xác định và thiết kế dữ liệu vật lí được chuẩn bị trong giai đoạn thiết kế trong.

(2) Phân hoạch và cấu trúc chức năng

Các chức năng được định nghĩa qua việc phát triển các hệ thống con trong giai đoạn thiết kế ngoài phải được xác định và thiết kế chương trình cho từng thao tác phải được chuẩn bị.

① Phân hoạch chức năng

Bước 1: Tại sao: Xác định các hệ con
Tại sao từng hệ con được phân chia trong thiết kế ngoài lại phải được làm rõ ràng.
Bước 2: Cái gì: Xác định các chức năng của hệ con

- Chức năng của từng hệ con được chia ra thành nhiều chức năng.
- Các chức năng được gộp nhóm.
- Xác định một chức năng mà từng nhóm thực hiện, hay một chủ đề cho từng nhóm.

Bước 3: Thế nào: Xác định chương trình
Thủ tục cho việc phát triển một chủ đề được xác định.

② Cấu trúc chức năng

Bước 1: Các chức năng chi tiết của một chương trình được kiểm điểm.
Các chức năng của chương trình được kiểm điểm.
Bước 2: Giao diện giữa các chương trình.
Dữ liệu truyền giữa chương trình được làm rõ.
Bước 3: Xác định luồng xử lí.
Thủ tục xử lí chương trình được xác định.

(3) Thiết kế dữ liệu vật lí

Trong bước này của thủ tục thiết kế trong, các đường truy nhập vào cơ sở dữ liệu và tệp, việc soạn thảo các tệp cũng như cách bố trí, được thiết kế dựa trên dữ liệu của thiết kế dữ liệu logic đã được tạo ra trong tiến trình thiết kế ngoài.

<Thủ tục thiết kế dữ liệu vật lí>

1. Phân tích các đặc trưng dữ liệu
2. Xác định cấu trúc tổ chức logic của tệp
3. Xác định cấu trúc tổ chức vật lí của tệp
4. Xác định phương tiện ghi nhớ dữ liệu
5. Thiết kế bố trí khoản mục dữ liệu

(4) Thiết kế vào-ra chi tiết

Trong bước này, các cái vào và cái ra dữ liệu chi tiết được thiết kế bằng việc dùng các khuôn mẫu chuyên dụng (sơ đồ không gian và các khuôn mẫu khác) dựa trên hình ảnh đã được chuẩn bị trong tiến trình thiết kế ngoài với xem xét thêm về hạn chế phần cứng.
Giao diện người dùng đồ hoạ (GUI) được đưa vào trong công việc thiết kế này từ nhiều năm trước. Bộ đọc kí tự quang học (OCR) và bộ đọc dấu hiệu quang (OMR) vẫn còn được dùng rộng rãi để đơn giản hoá công việc vào dữ liệu.
Việc đưa ra dữ liệu dùng dạng chuyên dụng phải được thiết kế khác nhau từ việc dùng các dạng phổ dụng.

<Các điểm cần lưu ý khi thiết kế cái vào và cái ra chi tiết>

- Sự phù hợp của nhiệm vụ vào hay ra dữ liệu với nhiệm vụ khác.
  Cái vào và cái ra phải được thiết kế có xem xét tới việc dễ thực hiện cho từng nhiệm vụ.
- Chọn phương tiện có tính đến việc làm dễ dàng xử lí và quản lí dữ liệu.
  Các phương tiện đưa vào kể cả bộ đọc kí tự quang học, đĩa mềm, bộ đọc mã vạch v.v..
  Phương tiện đưa ra bao gồm máy in, màn hình v.v.. Phương tiện đúng phải được chọn có tính tới sự thuận tiện cho xử lí và quản lí dữ liệu.
- Tính tới các hạn chế liên quan tới phần cứng.
  Cần nêu ra rằng máy in va đập có chức năng sao chép, nhưng nó lại ồn và tốc độ chậm, hay máy đọc kí tự quang và máy đánh dấu kí tự quang bị phụ thuộc phần cứng do lí do thiết kế. Những hạn chế liên quan tới phần cứng như vậy cần phải được tính tới.
- Giải quyết lỗi của dữ liệu vào.
  Điều mấu chốt là phải kiểm tra lỗi của dữ liệu đưa vào. Phải xác định cách kiểm tra và sửa lỗi.
- Đảm bảo dễ bảo trì, kể cả kiểm soát đúng đắn về phương tiện vào và ra.
  Thiết kế phải được chuẩn bị có xem xét tới việc dễ bảo trì và kiểm soát đúng phương tiện vào và ra.

(5) Tạo ra tài liệu thiết kế trong

Tài liệu trong được tạo ra như một sản phẩm của công việc được nêu ở trên:

<Nội dung của tài liệu thiết kế trong>

- Chính sách thiết kế trong
- Cấu hình hệ thống
- Chức năng chương trình
- Bố trí màn hình
- Bố trí cái vào và cái ra
- Tổ chức tệp
- Hiệu năng hệ thống
- Kế hoạch kiểm thử tích hợp

(6) Kiểm điểm thiết kế

Mục đích của kiểm điểm thiết kế trong giai đoạn thiết kế trong là:

- Làm hợp lệ rằng yêu cầu của người dùng được thoả mãn
- Kiểm chứng tính nhất quán với thiết kế ngoài được duy trì và dữ liệu thiết kế trong đã chuẩn bị có thể được trao cho giai đoạn thiết kế chương trình.

Ghi nhớ hai mục đích này, việc kiểm điểm thiết kế được tiến hành theo cách kĩ lưỡng. Cuộc họp kiểm điểm thiết kế là quan trọng trong bất kì giai đoạn nào; thiết kế chương trình và các tiến trình tiếp theo sau đó bị ảnh hưởng lớn bởi liệu cuộc kiểm điểm thiết kế tiến hành cho thiết kế trong có thành công hay không.

<Các điểm cần lưu ý khi tiến hành kiểm điểm thiết kế>

- Các sản phẩm được tạo ra ở giai đoạn thiết kế trong được kiểm chứng..
- Nội dung của thiết kế trong được kiểm điểm để xác nhận rằng nó là nhất quán với nội dung của tài liệu thiết kế ngoài, và rằng tất cả các chức năng được nêu ra trong tài liệu thiết kế ngoài là được cài đặt.
- Các chức năng được cài đặt được kiểm điểm để xác nhận rằng chúng được phân chia và cấu trúc thích hợp.
- Các giao diện giữa các chương trình được kiểm điểm để xác nhận rằng chúng đã được thiết kế đúng.
- Sự thích hợp của thiết kế dữ liệu vật lí được kiểm điểm.
- Các cái vào và cái ra chi tiết được kiểm điểm để xác nhận rằng chúng được thiết kế có tính tới việc làm dễ dàng cho người dùng.
- Các chức năng thiếu và những điểm không thích hợp được chỉ ra.

3.2 Phân hoạch và cấu trúc chức năng

Phân hoạch và cấu trúc chức năng là công việc phân hoạch các chức năng của các hệ con thành các chương trình sau khi xem xét tài liệu thiết kế ngoài và làm sáng tỏ các chức năng của từng chương trình, luồng các thao tác bao gồm nhiều chương trình, và sự phù hợp của chương trình nọ với chương trình kia.
Tiến trình cấu trúc là không thể thiếu được cho cả thiết kế ngoài và phát triển hệ thống.

3.2.1 Các đơn vị của việc phân hoạch và cấu trúc chức năng

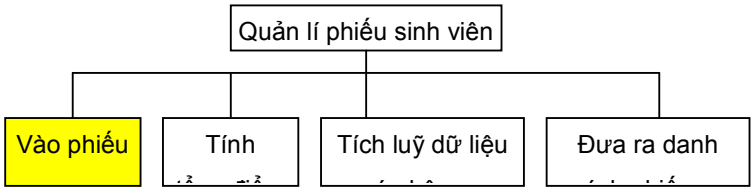
Trong giai đoạn thiết kế ngoài, hệ con được phân hoạch thành các chức năng. Trong giai đoạn thiết kế trong, hệ con được phân hoạch thành các chương trình dựa trên các chức năng của nó. Sự khác biệt giữa chức năng và chương trình được mô tả như sau:

(1) Chức năng và chương trình

<Thiết kế ngoài>

Hệ con được xây dựng lên trong việc phân hoạch nó thành các chức năng. Trong khi làm việc phân hoạch, mỗi chức năng được xét như đơn vị được phân hoạch nhỏ nhất; không kể đó là chương trình hay công việc thủ công.

Hình 3-2-1 Phát triển hệ con trong thiết kế ngoài

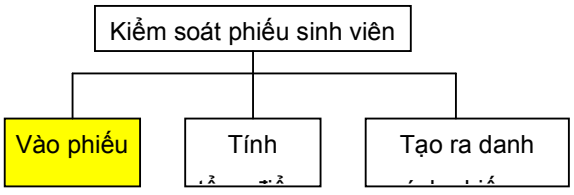


(thủ công)

<Thiết kế trong>

Các chức năng của hệ con được nghiên cứu kĩ lưỡng và chúng được phân hoạch ra, ứng với trong chương trình là một đơn vị của việc xử lí máy tính.

Hình 3-2-2 Phân hoạch và cấu trúc chức năng trong thiết kế trong



(2) Sắp thứ tự

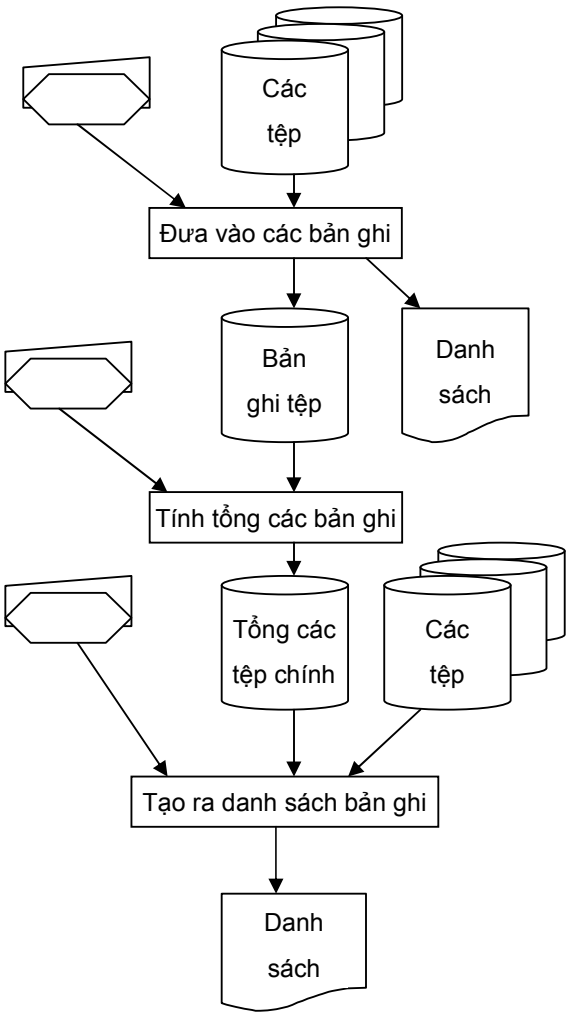
<Thiết kế ngoài>

Hệ con được phát triển, nhưng các chức năng của hệ con còn chưa được sắp thứ tự.

<Thiết kế trong>

Thứ tự thực hiện chương trình được xác định để cài đặt các chức năng và để thực hiện các thao tác một cách có hiệu quả. Nó thường được biểu diễn dưới dạng lưu đồ tiến trình (xem Hình 3-2-3).

Hình 3-2-3 Lưu đồ tiến trình



3.2.2 Các thủ tục phân hoạch và cấu trúc chức năng

Các thủ tục phân hoạch và cấu trúc chức năng được giải thích ở đây bằng việc tham chiếu tới bài tập được nêu sau đây.

<Bài tập> Tính lương tháng

Bản kê lương, bản báo cáo trả lương, và một tệp trả lương hàng năm được cập nhật trên cơ sở công việc của từng nhân viên và dữ liệu lương.

1. Giờ làm thêm và giờ vắng mặt trong từng tháng của nhân viên được đưa vào dựa trên dữ liệu làm việc.
2. Dữ liệu vào được kiểm tra. Nếu có chữa lỗi thì bản in danh sách lỗi sẽ được in ra. Trong trường hợp này, dữ liệu vào được sửa đổi và dữ liệu này được nạp lại vào máy.
3. Sau khi sửa xong lỗi, dữ liệu làm việc được sắp xếp tương ứng với mã phòng ban và nhân viên.
4. Các khoản phụ phí và số tiền chiết khấu được tính toán dựa trên tệp lương chính, và các tính toán khác được thực hiện.
5. Bản kê lương và báo cáo trả lương tháng được in ra dựa trên tệp kê lương có chứa dữ liệu thu được từ các bước 3 và 4 trên.
6. Tệp lương hàng năm được cập nhật dựa trên tệp kê lương.

(1) Nghiên cứu kĩ các chức năng

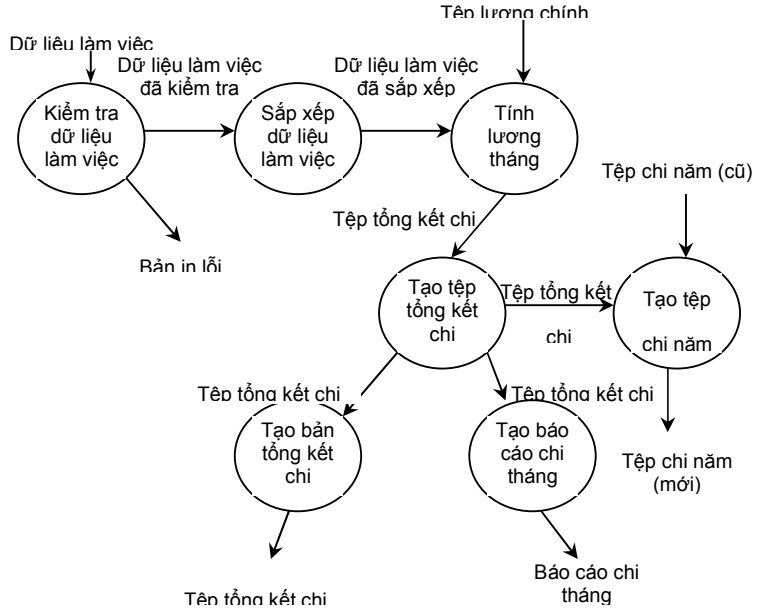
Các chức năng được nghiên cứu kĩ dựa trên lưu đồ làm việc mới (DFD vật lí mới) và lưu đồ con làm việc. Tất cả các chức năng phải được cài đặt trong các hệ con đều được nghiên cứu kĩ lưỡng. Trong trường hợp thiết kế ngoài, công việc thủ công mà máy tính không thực hiện sẽ được đưa vào các chức năng. Tuy nhiên, các chức năng được thảo luận trong mục này phải được cài đặt thành chương trình.
Trong trường hợp đặc biệt này, có thể quan niệm bẩy chức năng được vẽ trong Hình 1-2-4.

Hình 3-2-4 Nghiên cứu kĩ các chức năng

(2) Làm rõ luồng dữ liệu

Để phân hoạch chức năng, luồng dữ liệu cần được xử lí phải được xác định rõ. Luồng dữ liệu được biểu diễn bằng việc dùng dữ liệu logic dưới dạng biểu đồ luồng dữ liệu (DFD) hay sơ đồ bọt.

Hình 3-2-5 Làm rõ luồng dữ liệu (biểu diễn nó dưới dạng sơ đồ bọt)

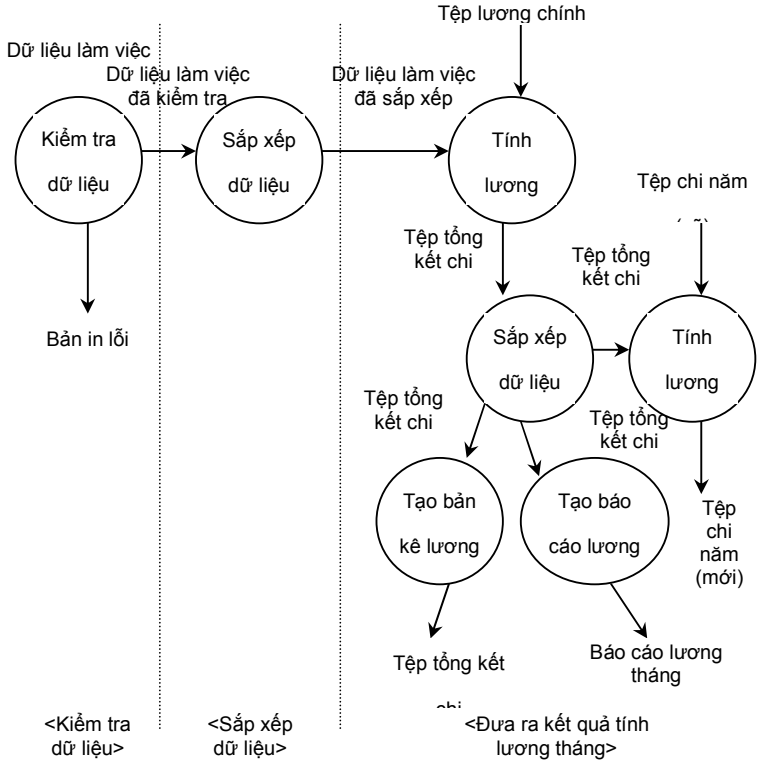


(3) Gộp nhóm chức năng

Dựa trên luồng dữ liệu được vẽ ở trên, các chức năng yêu cầu được gộp nhóm lại để cho phép máy tính thực hiện từng nhiệm vụ đã nêu. Bước này rất có thể được gọi là bước chuẩn bị, điều sẽ dẫn tới việc tạo ra và dùng lại các bộ phận..

Các chức năng trong luồng được vẽ theo (2) trên có thể được gộp nhóm như trong Hình 3-2-6.

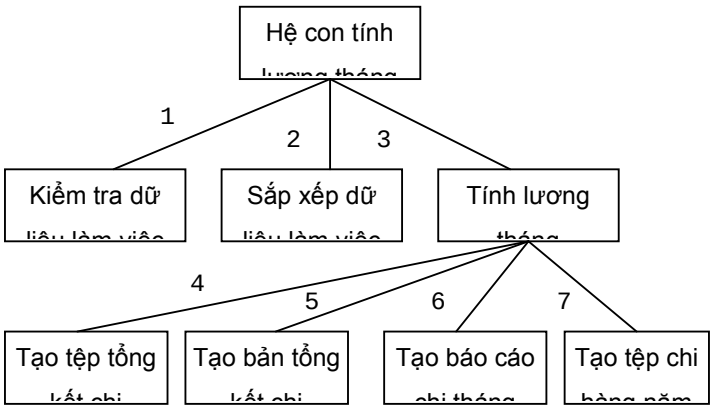
Hình 3-2-6 Gộp nhóm chức năng



(4) Cấu trúc phân cấp

Với chú ý được nêu cho luồng dữ liệu, các chức năng được yêu cầu sẽ được đưa vào một cấu trúc phân cấp trong các giai đoạn. Như được vẽ trong Hình 3-2-7, một sơ đồ giao diện chương trình với chương trình (bảng tham biến) được chuẩn bị để tạo mạch cho luồng dữ liệu vào và ra trong chương trình.

Hình 3-2-7 Cấu trúc phân cấp và giao diện chương trình với chương trình



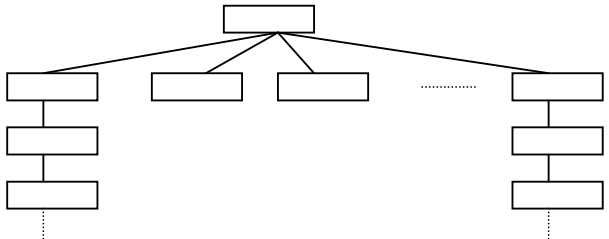
< Bảng tham biến >

	Cái vào	Cái ra
1	Dữ liệu làm việc	Dữ liệu làm việc đã kiểm tra. Danh sách lỗi
2	Dữ liệu làm việc đã kiểm tra	Dữ liệu làm việc đã sắp xếp
3	Dữ liệu làm việc đã sắp xếp Tệp chi chính	Tổng kết chi, Báo cáo chi tháng Tệp chi năm (mới)
4	Dữ liệu làm việc đã sắp xếp Tệp chi chính	Tệp tổng kết chi
5	Tệp tổng kết chi	Bản tổng kết chi
6	Tệp tổng tết chi	Báo cáo chi tháng
7	Tệp tổng kết chi Tệp chi năm (cũ)	Tệi chi năm (mới)

<Những điểm cần lưu ý khi xây dựng các chức năng trong cấu trúc phân cấp>

- Nên tránh phân cấp sâu. Nếu phân cấp được tạo ra quá sâu thì vùng thiết kế chương trình có thể bị bỏ qua.
- Nên thận trọng để giữ số chương trình trong một phân cấp là tối thiểu.

Hình 3-2-8 Ví dụ về cấu trúc phân cấp không đúng



(5) Xác định chức năng chương trình

Với hệ thống phân cấp ở cấp thấp nhất được vẽ ra từ (4), chi tiết về các trình con của chương trình được xác định với việc chú ý tới luồng dữ liệu. Nếu có thể được thì các bộ phận của phần mềm có thể được lắp ráp tại điểm này.

Hình 3-2-9 Xác định chức năng chương trình

Kiểm tra dữ liệu làm việc
Dữ liệu được đưa vào và kiểm tra. Nếu dữ liệu làm việc chứa lỗi, thì nó được đưa ra bản in lỗi Dữ liệu không chứa lỗi được ghi lên tệp trên bộ nhớ phụ xem như dữ liệu làm việc đã kiểm tra. Dữ liệu đưa ra bản in lỗi được sửa lại, và lại được đưa ra như dữ liệu làm việc đã kiểm tra
Dữ liệu làm việc đã sắp xếp
Dữ liệu làm việc đã kiểm tra được phân loại theo mã phòng ban và nhân viên để tạo ra dữ liệu làm việc đã sắp xếp (dữ liệu tháng)

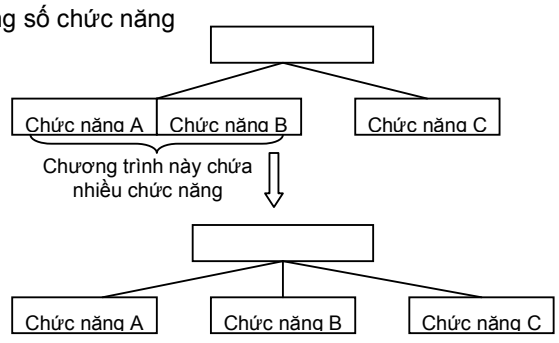
(6) Đánh giá kết quả của phân hoạch

Các chức năng đã phân hoạch được xém xét như cấu trúc chương trình và chúng được tính như các chương trình. Kết quả của công việc này, được mô tả theo (5) trên, được kiểm điểm lại.

<Đối tượng đánh giá kết quả của phân hoạch>

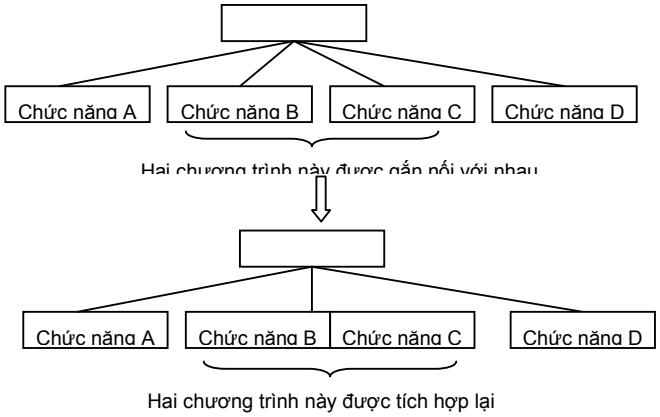
① Nếu chương trình chứa hai hay nhiều chức năng (tăng số của chức năng)

Hình 3-2-10 Tăng số chức năng



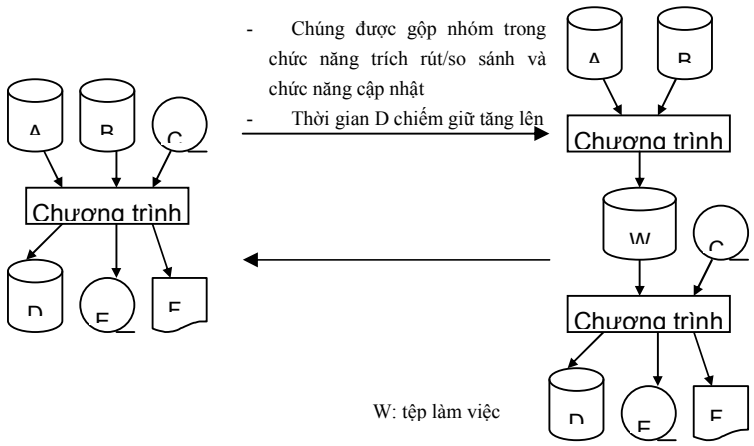
② Nếu các chương trình đã phân hoạch được gắn nối với nhau (tích hợp chức năng)

Hình 3-2-11 Tích hợp chức năng



③ Nếu cần xem xét lại các chương trình đã phân hoạch từ khía cạnh hiệu quả xử lí

Hình 3-2-12 Từ khía cạnh của tính hiệu quả xử lí



Sự độc lập của các chức năng được dùng như một phương tiện để đánh giá sự thích hợp của phân cấp và các chức năng được hàm chứa. Trong việc đánh giá sự độc lập của các chức năng, hai chỉ báo được nêu dưới đây sẽ được sử dụng. Chúng sẽ được giải thích chi tiết trong mục 4.2.3.

- Số chức năng: Số chức năng nên được tăng nhiều nhất có thể được.
- Gắn nối chức năng: Gắn nối chức năng nên được giảm nhiều nhất có thể được.

(7) Làm tài liệu đặc tả chức năng

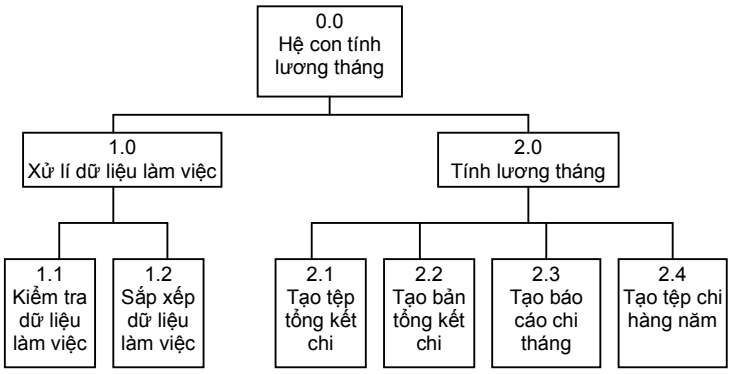
Các chức năng đã phân hoạch thành các chương trình được mô tả trong các tài liệu. Những tài liệu này tạo nên một phần của tài liệu thiết kế trong. Chúng có thể được chuẩn bị dưới dạng biểu đồ luồng dữ liệu, lưu đồ, hệ phân cấp cộng với cái vào xử lí cái ra (hierarchy plus input process output - HIPO), hay bằng bất kì phương tiện nào được chỉ định làm chuẩn nội

bộ.

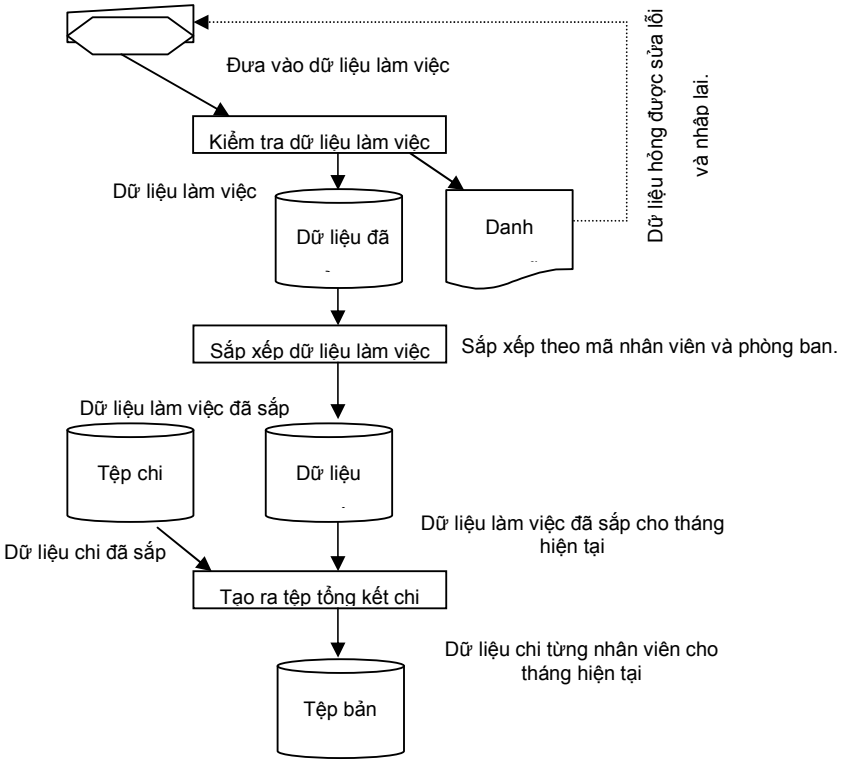
<Tài liệu>

Hình 3-2-13, 3-2-14 và 3-2-15 đưa ra cấu trúc hệ con, giao diện chương trình với chương trình và các chức năng chương trình tương ứng trong trường hợp vừa trình bầy ở trên.

Hình 3-2-13 Cấu trúc hệ con



Hình 3-2-14 Giao diện chương trình với chương trình



Hình 3-2-15 Chức năng chương trình (được tạo ra cho từng chương trình)

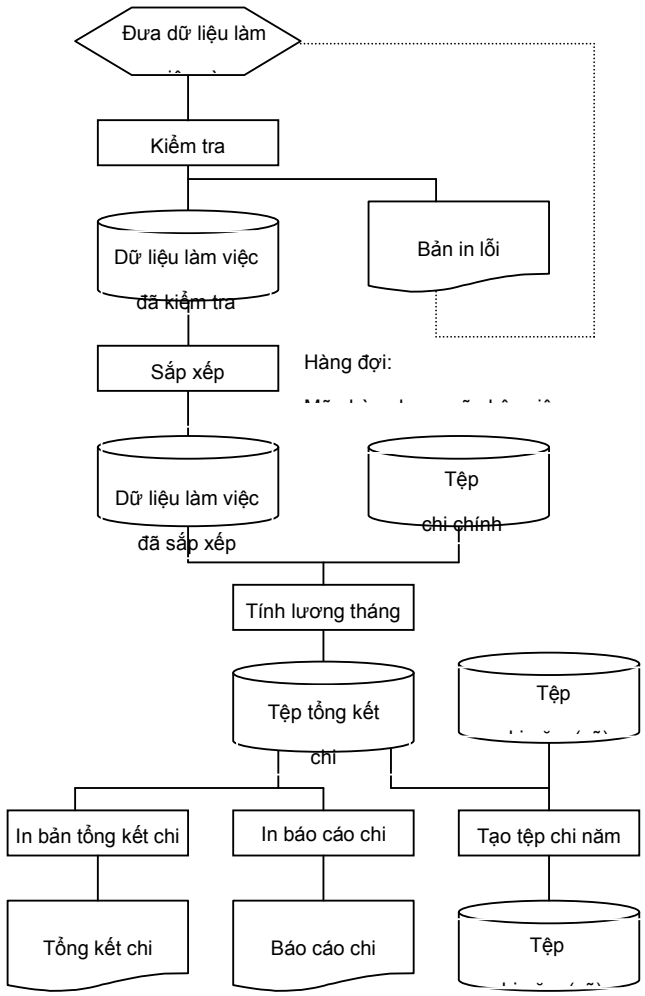
	Chi tiết về nhiệm vụ Cách thực hiện nhiệm vụ trong chương trình Giải thích các tham biến vào và ra Giải quyết lỗi Danh sách các thông báo Lưu ý đặc biệt

3.2.3 Phương pháp thiết kế có cấu trúc

Các kĩ thuật hay công cụ được dùng cho thiết kế có cấu trúc là như sau:
- Lưu đồ - Flowchart
- Biểu đồ luồng dữ liệu - Data flow diagram (DFD)
- Cấp bậc cộng với cái vào xử lí cái ra - Hierarchy plus input process output (HIPO)
- Sơ đồ có cấu trúc - Structured chart
- Biểu đồ chuyển trạng thái - State transition diagram
- Biểu đồ bọt - Bubble chart

(1) Lưu đồ

Lưu đồ cũng còn được gọi là sơ đồ tiến trình. (Lưu đồ thường được dùng cho các xử lí hạ lưu.)
Lưu đồ được tạo ra bằng việc dùng các kí hiệu được chuẩn hoá, bằng việc xác định và phân tích các vấn đề phức tạp và các thủ tục xử lí nhiệm vụ.
Hình 3-2-16 Lưu đồ (sơ đồ tiến trình)



(2) Biểu đồ luồng dữ liệu (DFD)

Biểu đồ luồng dữ liệu (DFD) được dùng để diễn đạt luồng xử lí nhiệm vụ hay thao tác hệ thống một cách có hệ thống. Trong khi viết biểu đồ luồng dữ liệu, nói chung cần phải chú ý tới nơi dữ liệu được dùng, cách nó được xử lí và nơi nó được lưu giữ. Biểu đồ luồng dữ liệu là một kĩ thuật dễ dùng và hiệu quả trong việc trình bày một tổng quan về yêu cầu người dùng. Gần đây nó đã được dùng để chỉ ra luồng dữ liệu trong hệ con.

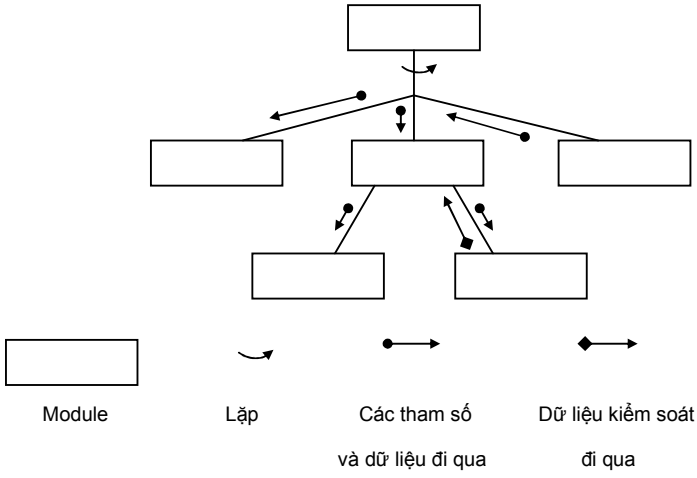
(3) Hệ phân cấp với cái vào xử lí cái ra (HIPO)

Hệ phân cấp với cái vào xử lí cái ra (HIPO) là một công cụ hỗ trợ làm tài liệu, được dùng như một phương tiện phụ để hỗ trợ cho công việc thiết kế.
Bởi vì nó phân chia hệ thống hay các chức năng chương trình một cách có hệ thống thành các phần theo thứ tự tuần tự, nên có thể đáp ứng cho nhiều nhu cầu đa dạng của người quản lí, người thiết kế, các phòng ban dùng chương trình, nhân viên chịu trách nhiệm phát triển hay bảo trì,v.v.

(4) Sơ đồ có cấu trúc

Sơ đồ có cấu trúc được dùng để biểu diễn cho các chức năng của từng chương trình theo một cách thức dễ hiểu. Việc dùng một sơ đồ có cấu trúc, các quan hệ chủ - tớ giữa các chương trình có thể được diễn tả như một cấu trúc phân cấp.
Một sơ đồ có cấu trúc rất dễ hiểu khi được dùng để biểu diễn cho các giao diện giữa các chương trình chứa hệ thống hay cấu trúc của từng chương trình.

Hình 3-2-17 Hình 3-2-17 Các kí hiệu được dùng trong sơ đồ có cấu trúc



(5) Biểu đồ chuyển trạng thái

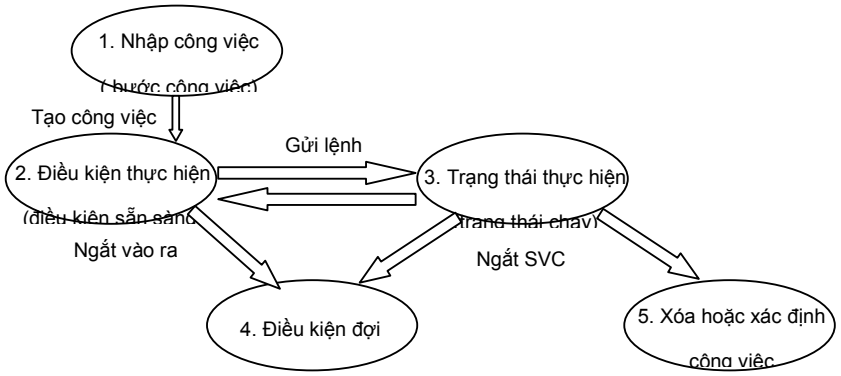
Biểu đồ chuyển trạng thái được dùng để chỉ cách trạng thái thay đổi (chuyển đổi). Nó thích hợp cho việc tổ chức hay diễn đạt trạng thái của hệ điều hành hay chương trình kiểm soát truyền thông. (Xem Hình 3-2-18.)

<Cách viết biểu đồ chuyển trạng thái>

- Các trạng thái có thể xảy ra được bao trong hình tròn hay hình chữ nhật.
- Các trạng thái được nối bằng mũi tên theo thứ tự chúng xảy ra.

- Các điều kiện gây cho một trạng thái đặc biệt xảy ra được viết sát cạnh mũi tên.

Hình 3-2-18 Biểu đồ chuyển trạng thái



(6) Sơ đồ bọt

Sơ đồ bọt được dùng để phân tích hay phân chia một hệ thống hay chương trình mong muốn. Các điểm cần lưu ý là khi phân tích một hệ thống hay chương trình:

- Dữ liệu nào được sinh ra ở đâu?
- Luồng dữ liệu như thế nào?
- Dữ liệu được xử lí như thế nào?

Công việc của thiết kế có cấu trúc có thể được sắp xếp hợp lí bằng việc dùng các kĩ thuật hay công cụ (1) tới (6) theo cách tổ hợp.

3.3 Thiết kế dữ liệu vật lí

Trong tiến trình thiết kế dữ liệu vật lí, các khoản mục và các đặc trưng (việc sử dụng và tăng tỉ lệ) của dữ liệu trong cơ sở dữ liệu, tệp hay cấu trúc bảng trong bộ nhớ, được xác định trong tiến trình thiết kế dữ liệu logic, sẽ được kiểm điểm và cách tổ chức vật lí và bố trí dữ liệu được thiết kế.
Dữ liệu và phương tiện ghi nhớ phải được xem xét có chú ý tới các đặc trưng tương ứng của chúng, và phương tiện lưu giữ thích hợp nhất phải được chọn lựa sao cho các đặc trưng dữ liệu có thể được dùng để tận dụng các ưu thế, các bản ghi có thể được tổ chức theo cách bố trí hiệu quả cao.

3.3.1 Thủ tục thiết kế dữ liệu vật lí

Thủ tục thiết kế dữ liệu vật lí được mô tả trong Mục 3.1.2. Mục này mô tả nội dung của công việc thiết kế dữ liệu vật lí, các điểm quan trọng cần lưu ý, và nhiều đặc trưng khác một cách thật chi tiết.

(1) Phân tích đặc trưng dữ liệu

Các đặc trưng của dữ liệu được phân tích chặt chẽ và thiết kế được chuẩn bị theo cách các đặc trưng của dữ liệu tương ứng có thể được tận dụng. Các điểm sau đây phải luôn ghi nhớ khi thiết kế:

<Điểm quan trọng cần lưu ý>

- Các đặc trưng và việc dùng dữ liệu
  - Đó là tệp chính hay tệp giao tác?
  - Phải duy trì nó bao lâu (duy trì thời gian dài hay duy trì tạm thời)?
  - Nó được dùng như dữ liệu dự phòng hay duy trì như bản ghi cập nhật?
- Thêm, xoá hay thay đổi dữ liệu
  - Khối lượng dữ liệu được thêm vào, xoá đi hay thay đổi trong thời kì xác định
  - Nội dung của nhiệm vụ cần được thực hiện (theo trật tự khoá hay ngẫu nhiên)
- Cập nhật
  - Nó được cập nhật hàng ngày, hàng tháng, hàng năm hay theo thời kì xác định?
- Cách dữ liệu được dùng
  - Nó được dùng cho xử lí theo lô?
  - Nó được dùng cho xử lí trực tuyến?
- Bảo trì
  - Dữ liệu có thể được khôi phục thế nào nếu nó bị phá huỷ?

(2) Xác định hệ thống tổ chức dữ liệu trong cấu trúc logic

Nếu phương tiện lưu giữ hay tổ chức vật lí được chấp nhận trước khi hệ thống dành cho việc tổ chức dữ liệu thành cấu trúc logic được xác định (tệp chính hay tệp giao tác, v.v.), thì tính hiệu quả của thao tác, sự thay đổi công việc thiết kế v.v.., sẽ bị ảnh hưởng bất lợi.

- Trường hợp chất nhận được

Hệ thống cho tổ chức dữ liệu thành cấu trúc logic được xác định trước hết, rồi tổ chức vật lí được xác định.
Cần: tệp hàng hoá phải được thiết kế như tệp chính.
Kết quả: Tệp hàng hoá được lưu giữ trên đĩa từ và được dùng như tệp tuần tự có chỉ số.

- Trường hợp không chấp nhận được
  Tổ chức vật lí được xác định trước, rồi hệ thống cho việc tổ chức dữ liệu thành cấu trúc logic được xác định.
  Cần: Các tệp phải được lưu giữ trên băng từ và được dùng như các tệp tuần tự có chỉ số.
  Kết quả: Tệp hàng hoá nên được dùng như tệp chính, nhưng băng từ là không tiện dùng.
  Chúng ta nên làm gì?

Điều cần khuyến cáo là hệ thống để tổ chức dữ liệu thành cấu trúc logic được xác định trước, rồi tổ chức vật lí mới theo sau.

<Điểm cần lưu ý>

- Phạm vi của việc dùng dữ liệu
  Dữ liệu có được dự định để dùng chỉ trong hệ thống đang được phát triển không? Hay nó phải được dùng trong một hệ thống khác như tệp chính? Hay nó được dùng chỉ cho lập trình?
- Dữ liệu tạm thời hay dữ liệu để cất giữ
  Dữ liệu được xử lí có chỉ được dùng tạm thời không? Hay nó phải được giải quyết như dữ liệu để cất giữ?
- Dữ liệu có được xử lí tuần tự không và nó có tăng dần hay tăng nhanh?
  Nó có phải là kiểu dữ liệu tăng dần và nó yêu cầu duy trì bản ghi cập nhật?

(3) Xác định phương tiện lưu trữ dữ liệu

① Xác định phương tiện ghi nhớ

Phương tiện ghi nhớ dữ liệu vật lí bao gồm:

- Đĩa từ (đĩa cứng)
- Băng từ
- Đĩa mềm
- Đĩa từ quang (MO)
- Ổ ZIP
- Streamer (để sao lưu dữ liệu lên đĩa cứng)

Đĩa từ-quang, ổ ZIP và streamer là các phương tiện ghi nhớ gần đây mới xuất hiện. Chúng có dung lượng nhớ lớn (hàng chục megabytes tới hàng trăm gigabytes) và thích hợp cho việc lưu giữ dữ liệu multimedia.
Các điểm sau đây nên được xem xét trong việc lựa phương tiện ghi nhớ thích hợp nhất:

- Dung lượng ghi nhớ
- Đặc trưng (phương pháp truy nhập dữ liệu)
- Tốc độ truy nhập
- Bảo trì, vận hành, giá cả, v.v..

Nếu bạn muốn truy nhập trực tiếp vào dữ liệu dùng khoá, thì chỉ những thiết bị ghi nhớ cho phép truy nhập trực tiếp, như đĩa từ, mới có thể được dùng.

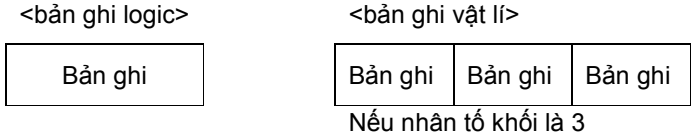
② Tính dung lượng nhớ và thời gian truy nhập

Trong việc tính dung lượng nhớ và thời gian truy nhập, cần phải chú ý tới sự khác biệt giữa các bản ghi logic và vật lí, và nhân tố khối.

<Kiểu bản ghi>

- Bản ghi logic: Một đơn vị dữ liệu được một chương trình xử lí; nó là một trong những sự vật được giải quyết bằng thiết kế trường.
- Bản ghi vật lí: Một đơn vị dữ liệu được đọc vào và được ghi ra phương tiện ghi nhớ

Hình 3-3-1 Bản ghi logic và vật lí



Nếu nhân tố khối là 3

<Điểm cần lưu ý>

- Nhân tố khối 3 nghĩa là một khối chứa ba bản ghi logic.
- Chương trình giải quyết với bản ghi logic, trong khi dữ liệu đưa vào, và đưa ra từ phương tiện ghi nhớ là các bản ghi vật lí.
- Có lỗ hổng giữa các khối - interblock gap (IBG).

Hình 3-3-2 đưa ra cách tính dung lượng bộ nhớ của đĩa từ và thời gian truy nhập.

Hình 3-3-2 Tính dung lượng nhớ của đĩa từ và thời gian truy nhập

1. Dung lượng nhớ của đĩa từ

<Các thông số của đĩa từ>

Số cylinder trên 1 đĩa	800 cylinder
Số rãnh trên cylinder	19 rãnh
Dung lượng rãnh	24 000 byte

(Cách tính)

Dung lượng lưu trữ của đĩa từ có thể được tính như sau:

Dung lượng lưu trữ của rãnh x số rãnh x số cylinder

Ví dụ: 24 000 bye trên 1 rãnh x 19 rãnh trên cylinder x 800 cylinder trên 1 đĩa

= 364 800 000 byte trên 1 đĩa

2. Thời gian truy cập của đĩa từ

<Các thông số của đĩa từ>

Khả năng lưu trữ của rãnh	15 000 byte
Thời gian định vị đầu từ trung bình	25 mili giây
Tốc độ quay	3000 vòng /phút
Độ dài một bản ghi	15 000 byte

<Cách tính>

Để tính được thời gian truy cập, cần 3 yếu tố: thời gian định vị đầu từ trung bình, thời gian tìm kiếm trung bình và thời gian truyền dữ liệu. Vì trong bảng trên chỉ có thời gian định vị đầu từ trung bình nên ta phải tính thời gian tìm kiếm trung bình và thời gian truyền dữ liệu.

Thời gian tìm kiếm trung bình có thể tính dễ dàng qua tốc độ vòng quay của đĩa từ. Nếu tốc độ quay là 300 vòng / phút thì thời gian đĩa quay 1 vòng mất 20 mili giây. Vì thời gian tìm kiếm trung bình bằng một nửa thời gian đĩa quay một vòng nên thời gian tìm kiếm trung bình là:

20 mili giây / 2 = 10 mili giây.

Vì tốc độ quay là 3000 vòng / phút = 50 vòng / giây nên thời gian truyền dữ liệu là:

50 rãnh / giây x 15 000 byte / rãnh = 750 x 10^3 byte / giây

Mỗi bản ghi có thể truyền 15 000 byte nên tốc độ truyền dữ liệu là:

$$\frac{15 \times 10^3 \text{ byte}}{750 \times 10^3 \text{ byte / giây}} = 0.02 \text{ giây} = 20 \text{ mili giây}$$

Vậy thời gian truy cập là:

25 mili giây + 10 mili giây + 20 mili giây = 55 mili giây.

(4) Thiết kế cách bố trí bản ghi

Tiến trình thiết kế cách bố trí bản ghi là bố trí các khoản mục dữ liệu.

① Thiết kế Trường (khoản mục) (thiết kế bản ghi logic)

Trong việc chuẩn bị thiết kế trường, các điểm sau đây nên được lưu ý:

- Thứ tự các khoản mục
  - Các khoản mục nên được thu xếp tuần tự, bắt đầu với khoản mục khoá. Nếu khoá bao gồm hai hay nhiều trường, thì những trường này nên được bố trí kế tiếp.
  - Các khoản mục nên được thu xếp theo thứ tự tầm quan trọng từ cao tới thấp hay từ cao tới thấp của tần số sử dụng.
  - Các khoản mục nên được thu xếp theo cách dễ dùng.
- Kích cỡ trường và kiểu dữ liệu
  - Nên dùng định dạng ngày tháng thống nhất (yymmdd, yyyymmdd, v.v.). đây là chuẩn ngày tháng và giờ quốc tế cần phải được thiết kế sao cho bất kì phần tử nào (năm, tháng, ngày) trong định dạng ngày tháng cũng có thể được giải quyết một cách riêng biệt và đồng thời, tất cả các phần tử có thể được dùng như một nhóm.
  - Các khoản mục số nên là số thập phân đã đóng gói.
- Trường lề
  - Nên lập ra một vùng dự trữ để cung cấp việc mở rộng tương lai (để làm tăng số trường).
  - Các trường lề nên được dùng để đối sánh chiều dài của các bản ghi (sánh từ 80 byte hay 256 byte).
  - Trường lề được thiết lập theo hai cách:

a. Không gian tự do được thiết lập tại đầu cuối

b. Không gian tự do được thiết lập ngay sau một tệp được dự định dùng trong mở rộng tương lai. Không gian này nên có cùng kích cỡ như trường dữ liệu hiện thời.

- Cần tính tới việc dễ dùng mã hoá.

② Kiểu bản ghi

Có các kiểu bản ghi sau:

a. Bản ghi chiều dài cố định

Trong trường hợp bản ghi chiều dài cố định, từng bản ghi đều bao gồm một tệp có cùng chiều dài. Nếu trong kiểu tệp này có tạo khối, thì chiều dài của từng khối trở thành như nhau.

Bản ghi chiều dài cố định được dùng rộng rãi bởi vì đặc trưng dễ dùng của nó.

b. Bản ghi chiều dài biến thiên

Trong trường hợp bản ghi chiều dài biến thiên, chiều dài của từng bản ghi trong một tệp là không như nhau. Với một tệp được tạo ra dùng định dạng bản ghi chiều dài biến thiên, từng bản ghi được chứa trong tệp phải được gắn sẵn với một bộ mô tả bản ghi chỉ ra chiều dài của bản ghi. Khi các bản ghi có chia khối, thì từng khối phải được gắn với bộ mô tả bản ghi để chỉ ra chiều dài toàn thể của khối đó.

Hình 3-3-3 Bản ghi chiều dài biến thiên

c. Bản ghi chiều dài không xác định

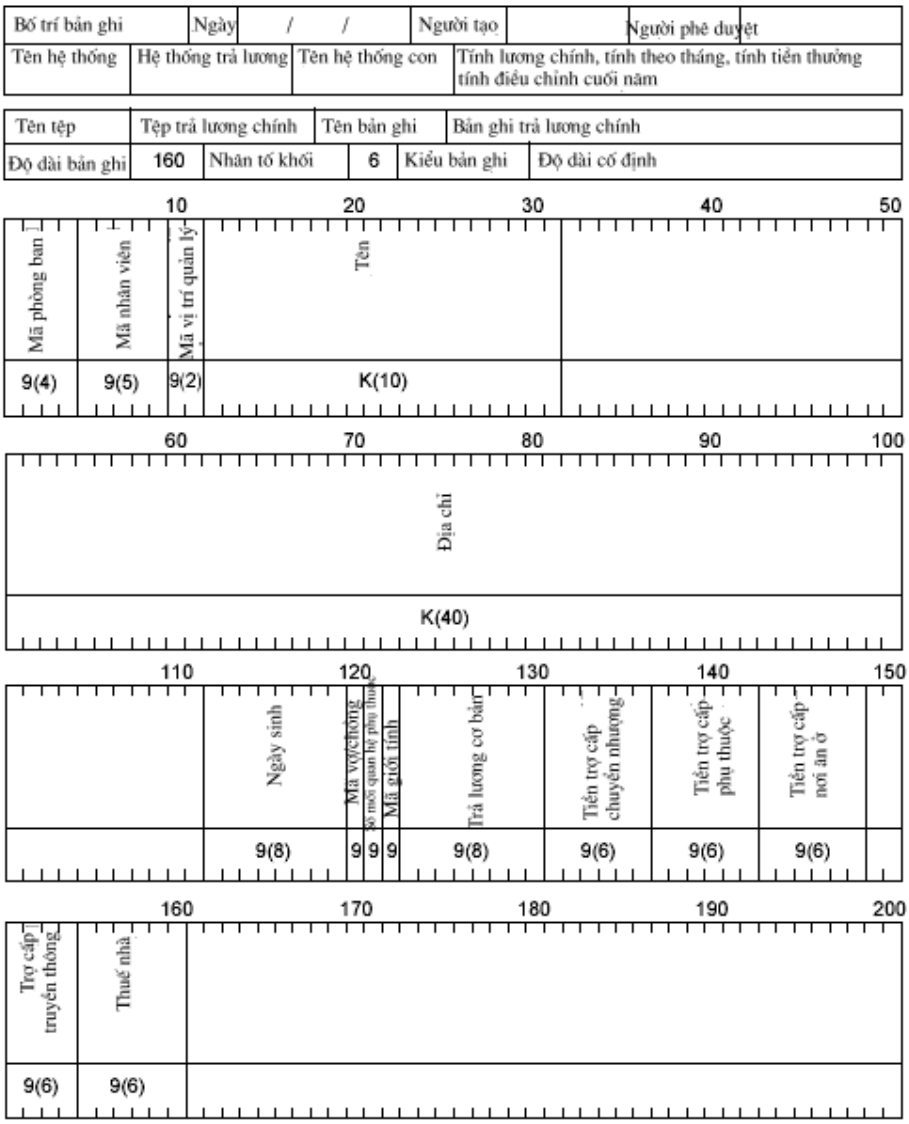
Trong trường hợp bản ghi chiều dài không xác định, chiều dài của từng bản ghi được xác định bởi chương trình, vì chiều dài của từng bản ghi logic một là khác nhau.

③ Bố trí bản ghi

Cần phải tính tới những điểm sau đây khi thiết kế bố trí bản ghi bằng cách dùng mẫu bố trí bản ghi (tệp):

- Với tên bản ghi, cần chọn dãy kí tự dễ nhớ biểu diễn cho nội dung của dữ liệu.
- X (kí tự chữ-số), 9 (số) và K hay G (kiểu Nhật) phải được đưa vào để chỉ ra kiểu của thuộc tính.
- Số các chữ số để tạo nên khoản mục dữ liệu phải được đưa vào. Một chữ số là một kí tự chữ-số (một byte). Trong trường hợp kiểu Nhật Bản, một kí tự tương ứng với hai byte, và do đó người ta khuyến cáo rằng số các chữ số cũng như số các kí tự được đưa vào để làm cho từng khoản mục dữ liệu dễ hiểu.

Hình 3-3-4 Ví dụ về bố trí bản ghi



3.3.2 Tổ chức dữ liệu vật lí

(1) Kiểu tổ chức tệp

Các kiểu tổ chức tệp vật lí được xác định dựa trên kết quả của phân tích đặc trưng dữ liệu, phương pháp tổ chức logic, chức năng chương trình, v.v...
Các kiểu tổ chức tệp vật lí là:

- Tệp tổ chức tuần tự
- Tệp tổ chức trực tiếp
- Tệp tuần tự chỉ số
- Tệp tổ chức được phân hoạch
- Tệp tổ chức ghi nhớ ảo - Virtual storage organization file (VSAM file)
- Bảng

Nên chọn kiểu tổ chức tệp vật lí thích hợp nhất theo mục đích đã cho.

① Tệp tổ chức tuần tự

Trong tệp tổ chức tuần tự, các bản ghi được lưu giữ tuần tự theo vị trí liên tiếp trên thiết bị ghi nhớ.
Các bản ghi thường được thu xếp theo thứ tự tăng hay giảm với khoản mục dữ liệu nào đó được chỉ định làm khoá, hay theo thứ tự các bản ghi được tạo ra.

<Đặc trưng>

- Kiểu tệp này có thể được tạo ra trên bất kì phương tiện nhớ nào, đĩa từ, băng từ v.v.
- Việc đọc và ghi tệp bắt đầu từ đầu bản ghi; không hỗ trợ cho truy nhập ngẫu nhiên.
- Bởi vì các bản ghi được lưu giữ một cách vật lí và kế tiếp nhau, nên bản ghi mới không thể được bổ sung hay chèn thêm vào. Do đó, để cập nhật tệp nào đó trên băng từ, tệp mới phải được tạo ra. Tuy nhiên trong trường hợp của đĩa từ, dữ liệu có thể được ghi lại vào tệp gốc.
- Tốc độ truy nhập nhanh.
- Kiểu tệp này là thích hợp cho xử lí theo lô hay sao lưu dữ liệu.

② Tệp tổ chức trực tiếp

Tệp tổ chức trực tiếp chỉ có thể được dùng trên thiết bị bộ nhớ truy nhập trực tiếp, như đĩa từ.

<Đặc trưng>

- Địa chỉ của dữ liệu được tính bằng việc dùng phương pháp tính địa chỉ đặc biệt gọi là phương pháp băm.
- Cùng một địa chỉ có thể xuất hiện nhiều lần khi việc chuyển đổi địa chỉ được thực hiện. Một địa chỉ như vậy được gọi là đồng nghĩa.
- Việc truy nhập trực tiếp được hỗ trợ. Trong một số trường hợp, truy nhập tuần tự cũng được hỗ trợ.
- Hiệu quả bộ nhớ là cao.
- Tốc độ xử lí nhanh.
- Kiểu tệp này là phù hợp với xử lí thời gian thực trực tuyến đòi hỏi tốc độ cao.

③ Tệp tuần tự có chỉ số

Tệp cho phép tham chiếu tới chỉ số vị trí lưu giữ dữ liệu cũng như việc đọc dữ liệu đặc biệt được gọi là tệp tuần tự có chỉ số. Kiểu tệp này chỉ có thể được dùng trên thiết bị nhớ truy nhập trực tiếp (direct access storage devices: DASD).

<Đặc trưng>

- Vùng bộ nhớ dữ liệu bao gồm một vùng chỉ số, vùng dữ liệu chính và vùng tràn.
- Vùng chỉ số bao gồm chỉ số chính, chỉ số trụ và chỉ số rãnh. Các chỉ số này được thiết kế để sánh với các khoá của từng bản ghi sao cho dữ liệu có thể được duyệt và tìm. Do đó, các giá trị khoá phải được thu xếp theo thứ tự tăng.
- Cả truy nhập tuần tự và trực tiếp đều là có thể.
- Vì vùng chỉ số phải được duyệt trước, nên mức hiệu quả lưu giữ là thấp.
- Tốc độ xử lí chậm.
- Kiểu tệp này phù hợp với tệp chính cho công việc giấy tờ.

④ Tệp tổ chức có phân hoạch

Trong tệp tổ chức có phân hoạch, tệp tuần tự được phân hoạch thành các tệp con gọi là thành viên, và một danh mục được tạo ra để chỉ ra vị trí bắt đầu của từng thành viên để cho từng thành viên lẻ có thể được truy nhập trực tiếp.
Kiểu tệp này chỉ có thể được dùng trên các thiết bị nhớ truy nhập trực tiếp (DASD), như đĩa từ, như trong trường hợp của tệp tổ chức trực tiếp.

<Đặc trưng>

- Mặc dầu từng thành viên có thể được truy nhập trực tiếp, các bản ghi được chứa trong một thành viên được truy nhập tuần tự.
- Tốc độ truy nhập gần như là giống tốc độ của tệp tổ chức tuần tự.
- Mức độ hiệu quả ghi nhớ thấp hơn mức độ hiệu quả của tệp tổ chức tuần tự vì các danh mục được tạo ra.
- Kiểu tệp này là thích hợp cho tệp chương trình và các thư viện khác nhau.

⑤ Tệp tổ chức lưu giữ ảo (tệp VSAM file)

Tệp tổ chức lưu giữ ảo có thể được dùng với hệ điều hành có chức năng bộ nhớ ảo.
Kiểu tệp này chỉ có thể được dùng trên các thiết bị lưu trữ truy nhập trực tiếp (DASD), như đĩa từ, như trong trường hợp của tệp tổ chức trực tiếp.

<Đặc trưng>

- Tệp này được thiết kế bằng việc tích hợp ba phương pháp truy nhập được dùng cho các tệp tổ chức tuần tự có chỉ số.
- Bởi vì kiểu tệp được kiểm soát bởi hệ điều hành, nên người phát triển hệ thống không cần nhớ chiều dài khối hay chiều dài bản ghi.
- Kiểu tệp này được chia thêm thành ba kiểu:

Tập dữ liệu tuần tự theo khoá : KSDS (tương đương với tệp tổ chức tuần tự có chỉ số)
Tập dữ liệu tuần tự theo việc đưa vào: ESDS (tương đương với tệp tổ chức tuần tự)
Tập dữ liệu bản ghi tương đối : RRDS (tương đương với tệp tổ chức trực tiếp)

⑥ Bảng

Bảng là một vùng làm việc được thiết lập trên đơn vị bộ nhớ chính. Trong trường hợp của COBOL, các mục sau được định nghĩa trong Data Division:

- Tên bảng
- Số các bảng
- Tên của khoản mục trong bảng
- Kiểu khoản mục dữ liệu (dữ liệu chữ-số, dữ liệu số, v.v..)

<Kí tự>

- Bởi vì bảng nằm trong bộ nhớ, nên tốc độ truy nhập rất nhanh.
- Cả hai việc truy nhập tuần tự và trực tiếp đều có thể có do cấu trúc chương trình.

- Nếu máy tính bị tắt, thì dữ liệu bị mất. Do đó, bảng không thể được dùng như tệp chính.
- Bảng nói chung được dùng như một vùng để thực hiện duyệt tốc độ cao trên dữ liệu, hay tính tổng dữ liệu.

(2) Cách xử lí

Cách dữ liệu được định dạng và duy trì có ảnh hưởng tới tính sử dụng được của hệ thống. Khi phát triển cấu trúc dữ liệu trong thiết bị nhớ, phải xem xét các nhân tố khác nhau, chẳng hạn phương pháp truy nhập, hiệu quả truy nhập, hiệu quả không gian, định vị chỗ hỏng, hiệu quả của khối lượng bộ đệm trong bộ nhớ, v.v.. Tính tới tất cả những nhân tố này, việc giải quyết các tệp và dùng cơ sở dữ liệu nên được xem xét tới.

Hiệu năng, giá cả, giải quyết trục trặc và những nhân tố khác liên kết với thiết kế hệ thống chung nên được xác định cùng với xem xét về các yêu cầu hệ thống. Tuy nhiên khó đạt tới tính hiệu quả và dung lượng nhớ, hay tính hiệu quả của việc duyệt (hiệu quả ghi) vào cùng lúc. Nhưng nhân tố liên quan tới tính hiệu quả này phải được xác định để cho chúng có thể được giữ cân bằng.

Các tệp cũng được diễn giải một cách khác nhau, tuỳ theo hệ điều hành mà trong đó chương trình phần mềm được tạo ra đang chạy. Cần kiểm chứng các chức năng dịch vụ quản lí tệp do hệ điều hành cung cấp. Tương ứng với diễn giải của UNIX, một trong những hệ điều hành chính, nội dung của tệp nên được chương trình nhận ra và không cần hệ thống tệp của UNIX nhận ra. Tuy nhiên theo diễn giải của MVS, một nhóm dữ liệu logic được nhận ra như một bản ghi và đơn vị cơ sở của thao tác tệp được người lập trình thiết kế, là được hệ thống tệp hỗ trợ.

3.4 Thiết kế vào ra chi tiết

Trong giai đoạn thiết kế ngoài, màn hình và báo cáo được thiết kế để cho phép người dùng nhận ra hình ảnh của màn hình và cái ra. Việc thu xếp chính xác các khoản mục, kích thước font, mầu sắc, kiểu biểu tượng và các chi tiết khác là chủ đề cho các ràng buộc liên quan tới phần cứng hay hệ thống không được tạo ra trong giai đoạn thiết kế ngoài.
Trong giai đoạn thiết kế trong, những chi tiết như vậy được dựa trên hình ảnh đã được tạo ra trong giai đoạn thiết kế ngoài. Thiết kế trong được chuẩn bị với xem xét liên quan tới các ràng buộc phần cứng. Bởi vì thiết kế trong dựa trên công việc của thiết kế ngoài, nên chủ điểm chính là để đạt tới việc dễ dùng cho người dùng.

3.4.1 Thiết kế dữ liệu vào chi tiết

Mẫu dữ liệu vào là một trong những dữ liệu đưa vào chính. Mục này mô tả thiết kế chi tiết cho mẫu dữ liệu vào.

(1) Mục đích thiết kế

Người dùng sử dụng mẫu dữ liệu vào. Để thiết kế mẫu dữ liệu vào được trau chuốt, có cân nhắc kĩ, cần ghi nhớ về các điều kiện của người dùng và tốc độ, độ chính xác và tính dễ dàng của dữ liệu.
Các mục đích phải đạt được trước hết trong thiết kế mẫu dữ liệu vào là:
- Dễ viết
- Dễ đưa vào máy tính (gõ vào)

(2) Thiết kế

Kích cỡ của mẫu, số lượng bản sao, mầu sắc được dùng trong mẫu và khuôn khổ của mẫu dữ liệu vào phải được xác định.

① Dễ đưa vào dữ liệu

- Phải giữ lượng công việc cần viết ở mức tối thiểu
  Những điểm sau nên được xem xét để tiết kiệm thời gian và lao động của người dùng trong việc đưa dữ liệu vào:
  - Phương pháp lựa chọn (người dùng được yêu cầu đưa ra chọn lựa và đóng khuôn một khoản mục)
  - Số tối thiểu các khoản mục đưa vào
  - Sự trùng lặp
  - Các khoản mục cố định cần được in ra trên mẫu đưa vào dữ liệu
  - Những khoản mục ít được đưa vào sẽ được gộp nhóm vào vùng lưu ý
- Khoản mục đưa vào
  Khoản mục đưa vào nên được thu xếp theo trình tự ý nghĩ.
  - Người dùng nên được hướng dẫn trước hết đưa dữ liệu vào những khoản mục cần chính.
  - Người dùng nên được hướng dẫn đưa dữ liệu vào từ góc trên bên trái tới góc dưới bên phải.

- Các khoản mục đưa vào nên dễ đọc và dễ hiểu.

Hình 3-4-1 Cách từng khoản mục có thể được làm cho dễ đọc và dễ hiểu

1. Dưa ra những gì mà người sử dụng muốn viết một cách rõ ràng

Ngày_____	Khó hiểu
Ngày: năm 2000__, tháng_____ngày__	Dễ hiểu

2. Hướng dẫn người sử dụng ngăn anh ta/ cô ta từ việc tạo ra lỗi nhập

Số lượng	Khó hiểu
Nợ đến hạn _____, _____, _____ (không bao gồm nợ đến hạn)	Điều này ngăn người sử dụng từ việc tạo ra lỗi nhập.

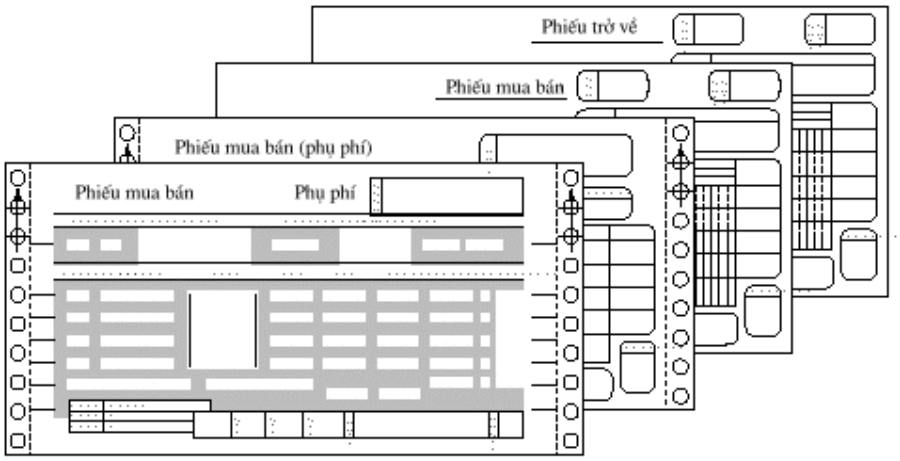
② Dễ đưa vào (từ bàn phím) máy tính

Độ chính xác của dữ liệu có thể được đảm bảo nếu dữ liệu được đưa vào nơi nó được sinh ra (tại chỗ làm việc). Do đó, mẫu đưa dữ liệu vào phải được thiết kế dựa trên mục đích của từng nhiệm vụ xử lí dữ liệu, và nội dung của các thao tác hệ thống mới (luồng dữ liệu) với giả thiết rằng nó được dùng tại từng chỗ làm việc.

- Nên dùng mầu như mẫu đưa vào dữ liệu để cho phép người dùng nhận ra nó ngay thoạt nhìn.
- Từng khoản mục dữ liệu nên được bao trong đường đậm để dễ thấy.
- Một khoản mục tách biệt nên được thiết lập để cho phép người dùng phân biệt trên mẫu đưa dữ liệu vào mà dữ liệu đã được đưa vào với những phần chưa đưa vào.

Mẫu đưa vào dữ liệu được nêu như sau.

Hình 3-4-2 Mẫu đưa vào dữ liệu (riêng cho OCR)



③ Thiết kế mẫu OCR

Bằng việc dùng bộ đọc kí tự quang học OCR, bạn có thể đưa dữ liệu vào trực tiếp bằng việc cho mẫu qua máy đọc. Với giai đoạn hiện tại của việc phát triển công nghệ OCR, việc đọc sai thường xuất hiện ngay cả dữ liệu được viết đúng và dễ đọc.

Có một số biện pháp thoả hiệp được chọn: chú ý tới việc làm rõ mẫu OCR (để ngăn cản việc đọc lầm) hay đưa ra những giải thích về kiểu kí tự.

(3) Kiểm tra dữ liệu đưa vào

Dữ liệu đưa vào bao giờ cũng phải được kiểm tra vì xác suất người dùng phạm sai lầm đưa vào là cao, hay vì máy OCR hay OMR đọc lầm dữ liệu vào. Sai lầm trong dữ liệu đưa vào có thể dẫn tới không chỉ kết quả không đúng, mà còn cả việc làm tắt hệ thống hay hệ thống chạy lầm.
Bản thân dữ liệu có thể chứa lỗi. Dữ liệu đưa vào vượt quá một giới hạn thời gian đưa vào xác định hay dữ liệu nào đó bị mất. Tất cả những vấn đề này đều được coi là lỗi đưa vào.

① Sửa lỗi dữ liệu

Khi một lỗi dữ liệu được tìm thấy, nó phải được sửa ngay lập tức. Những hành động cần thực hiện khi tìm thấy lỗi phải được xác định rõ ràng.

- Lỗi được tìm thấy trước khi dữ liệu được đưa vào máy tính
  Nếu lỗi được tìm thấy khi dữ liệu được đưa vào, thì mẫu đưa vào dữ liệu phải được trả lại cho bộ phận điền vào nó. Thao tác viên không bao giờ được sửa lỗi này. Nếu thao tác viên sửa nó theo chủ quan của mình, thì có thể một vấn đề trầm trọng, bất ngờ sẽ xảy ra về sau. Các lỗi được tìm thấy ở giai đoạn này có ảnh hưởng nhỏ tới thao tác xử lí dữ liệu.
- Lỗi do chương trình phát hiện ra
  Nếu lỗi được tìm thấy khi một chương trình đã loại bỏ dữ liệu đưa vào, thì việc xử lí dữ liệu đã được tiến hành. Do đó, phải xác định những điểm nào lỗi cần được sửa chữa.

② Giải quyết lỗi dữ liệu

- Bỏ qua lỗi dữ liệu và tiếp tục xử lí dữ liệu
  Nếu chỉ có vài lỗi dữ liệu trong hàng nghìn dữ liệu được xử lí với mục đích thống kê, thì những lỗi như vậy chỉ có ảnh hưởng tối thiểu lên các thao tác xử lí toàn thể, và có thể bỏ qua được. Nếu tất cả các dữ liệu phải được đăng kí, hay nếu tỉ lệ dữ liệu lỗi tăng lên, phương pháp giải quyết lỗi này không thể dùng được.
- Thực hiện xử lí dữ liệu sau khi tất cả các lỗi dữ liệu đã được sửa
  Phương pháp giải quyết lỗi này được dùng nếu nó được giải quyết trong điều kiện rất nghiêm ngặt và không một lỗi nào được phép xuất hiện (chẳng hạn dữ liệu đăng kí). Tuy nhiên công việc này bao gồm việc sửa lỗi dữ liệu rất tốn thời gian.
- Chỉ dùng dữ liệu đúng và tiếp tục xử lí dữ liệu
  Chỉ dùng dữ liệu đúng thì việc xử lí dữ liệu có thể được tiếp tục. Sau khi được hoàn tất, các lỗi dữ liệu được sửa, và được gộp với dữ liệu đúng. Tiến trình chính, bắt đầu khi tất cả các dữ liệu đã được làm thành sẵn có.

③ Phương pháp kiểm tra dữ liệu

Hình 3-4-3 đưa ra các phương pháp kiểm tra dữ liệu chính.
Hình 3-4-3 Các phương pháp kiểm tra dữ liệu

Phương pháp kiểm tra dữ liệu		Các vấn đề cần kiểm tra
Kiểm tra tính hợp lệ	Kiểm tra định dạng	Dữ liệu được tạo theo định dạng cụ thể
	Kiểm tra số	Các dữ liệu khác với các ký tự số không được nhập vòa trong các mục số
	Kiểm tra giới hạn	Dữ liệu không được quá phạm vi và giới hạn cho phép
	Kiểm tra vùng	Giá trị phải trong giới hạn cho phép
	Kiểm tra tràn	Dữ liệu không được vượt quá độ dài dữ liệu cho phép
	Kiểm tra chữ số kiểm tra	Các số kiểm tra phải được mã hóa để xác định các số đọc đúng
Kiểm tra tệp	Kiểm tra tổng	Kết quả tính toán bằng máy bàn phải được so sánh với kết quả tính toán bằng máy tính
	Kiểm tra chuỗi	Dữ liệu được sắp xếp theo chuỗi phù hợp với chuẩn cho phép
	Kiểm tra số dư	Các số bên có và bên nợ phải phù hợp
	Kiểm tra	Số dữ liệu thực tế phải phù hợp với số dữ liệu trong máy tính
Kiểm tra bằng mắt		Dữ liệu đưa ra phải được kiểm tra để tìm các lỗi dữ liệu

3.4.2 Thiết kế màn hình

Thiết kế màn hình là một trong những nhân tố quan trọng mà người dùng xem xét để xác định liệu hệ thống được thiết kế tốt hay tồi, như đã được giải thích trong mục dành cho phân tích hệ thống. Màn hình là khuôn mặt của hệ thống đối với người dùng. Màn hình phải được thiết kế với ưu tiên cao nhất dành cho việc dễ dùng.
Bởi vì các kĩ thuật giao diện người dùng đồ hoạ (GUI) dùng ngôn ngữ trực quan đã trở thành các kĩ thuật máy tính chủ đạo, nên mục này mô tả cho thiết kế màn hình dùng các kĩ thuật GUI.
Thiết kế màn hình với việc dùng các kĩ thuật GUI được thực hiện bằng việc làm bản mẫu. Thủ tục thiết kế màn hình được nêu dưới đây.

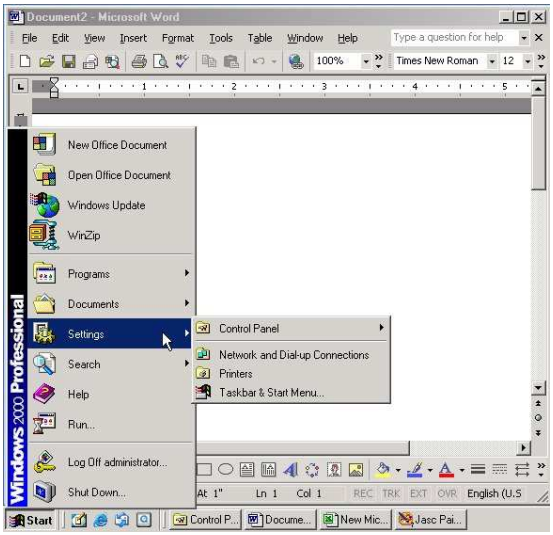
<Thủ tục thiết kế màn hình>

1. Bố trí chuẩn màn hình, các yêu cầu phải được đáp ứng để thực hiện các chức năng xác định, và mức độ thành thạo của người dùng được làm rõ ràng.
2. Màn hình bản mẫu được tạo ra bằng việc dùng ngôn ngữ trực quan.
3. Người dùng đánh giá màn hình bản mẫu. Các sửa đổi và cải tiến được thực hiện để hoàn chỉnh màn hình bản mẫu.

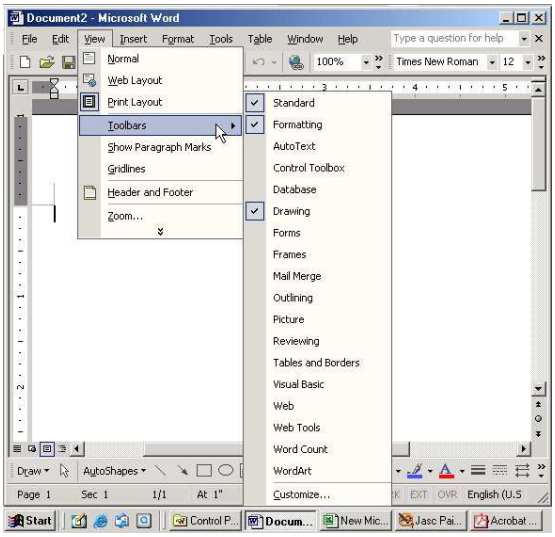
(1) Giao diện người dùng đồ hoạ (GUI)

GUI dùng các biểu tượng trực giác (kí hiệu hình ảnh), các menu bật ra, menu thả xuống v.v để cho phép các nhiệm vụ được thực hiện. Đặc biệt, một mũi tên được di chuyển bằng việc dùng thiết bị trỏ (chuột hay các thiết bị khác) để định vị vào chỗ mong muốn, nơi một biểu tượng hay khoản mục được chọn hướng dẫn cho máy tính thực hiện nhiệm vụ đã nêu.

Hình 3-4-4 Biểu tượng và menu bật ra



Hình 3-4-5 Menu thả xuống (kéo xuống)



Các chức năng GUI được xây dựng bên trong hệ điều hành của máy tính cá nhân. Tuy nhiên, trong trường hợp của UNIX, các chức năng GUI phải được cài đặt thêm. Các biểu diễn, ý nghĩa và các thao tác trên các biểu tượng là khác nhau tuỳ theo GUI được dùng. Bởi vì việc chuẩn hoá đã được thực hiện cho từng GUI, nên năng suất thiết kế đã cải tiến và số lỗi đã được giảm đi.

(2) Thuật ngữ liên quan tới môi trường GUI (Windows)

Trước khi mô tả về thiết kế màn hình, một số thuật ngữ liên quan tới thiết kế màn hình trong GUI được giải thích vắn tắt ở đây:

① Chương trình được điều khiển theo biến cố

Tất cả các ứng dụng được dùng trong môi trường GUI đều là các chương trình được điều khiển theo biến cố. Các ứng dụng được phát triển trước khi có GUI đều đã được thiết kế với lược đồ điều khiển tuần tự. Với chương trình điều khiển theo biến cố, các hành động

như bấm chuột hay rê con chạy có thể gây ra một biến cố đặc biệt, mà đến lượt nó lại kích hoạt các thủ tục định sẵn tương ứng.

② Windows (cửa sổ)

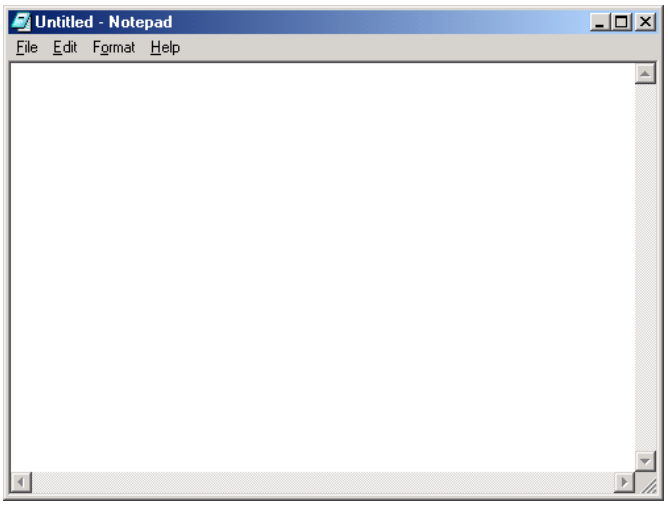
Cơ sở của GUI là các cửa sổ. Chúng thay đổi từ ứng dụng nọ sang ứng dụng kia.

③ Tài liệu

Nội dung được nêu trong một cửa sổ được gọi là tài liệu. Kiểu tài liệu được gọi là giao diện tài liệu và được phân loại đại thể thành hai kiểu:

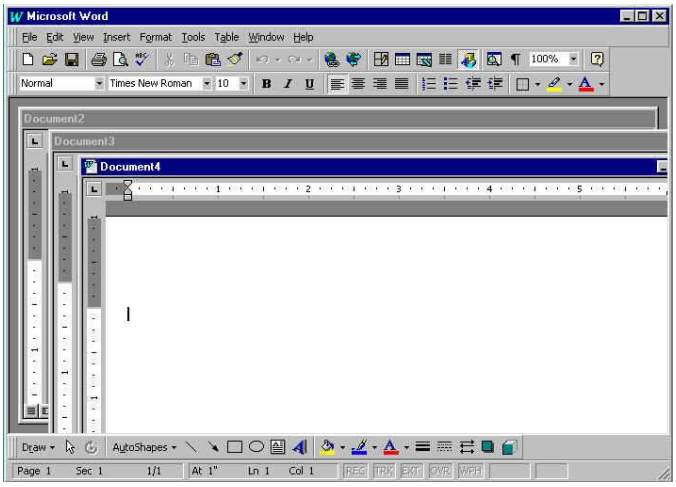
a. Giao diện một tài liệu (SDI)
Giao diện một tài liệu (SDI) nêu ra một tài liệu bên trong một ứng dụng. (Xem Hình 3-4-6.)

Hình 3-4-6 Giao diện một tài liệu (SDI)



b. Giao diện đa tài liệu (MDI)
Giao diện đa tài liệu (MDI) nêu ra nhiều tài liệu bên trong một ứng dụng.

Hình 3-4-7 Giao diện đa tài liệu (MDI)

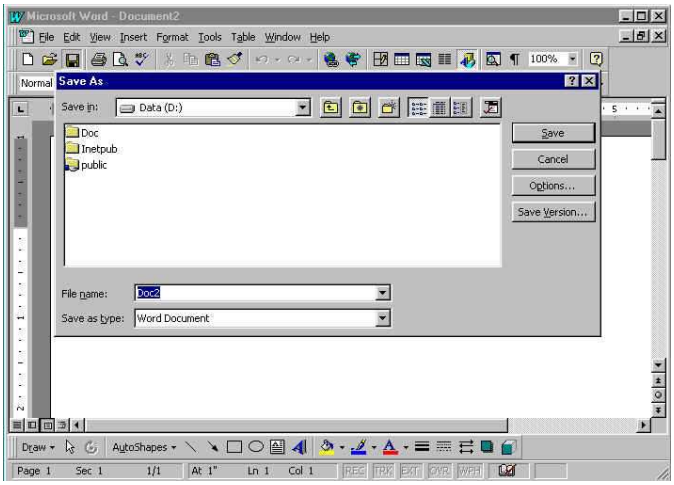


④ Hộp thoại

Hộp thoại được dùng để nêu ra thông tin phản hồi cho người dùng và lời nhắc người dùng

đáp ứng. (Xem Hình 3-4-8.)

Hình 3-4-8 Hộp thoại



(3) Thủ tục và nhiệm vụ cho thiết kế màn hình bằng việc dùng GUI

Thiết kế màn hình được tiến hành theo sáu bước dưới đây:

① Chuẩn hoá

Với việc bố trí trong các màn hình và tương tác, cần xác định các qui tắc chuẩn.

Hình 3-4-9 Chuẩn hoá màn hình

Phần tử được chuẩn hoá	Nội dung
Tính liên tác	Phạm vi vận hành được chỉ ra sao cho người dùng có thể chọn lựa và đưa ra hướng dẫn. Điều mong muốn là màn hình có tính năng thao tác trực tiếp cho phép đáp ứng ngay lập tức.
Tính đều	Tính nhất quán hay tính đều phải được đảm bảo theo cách cùng chức năng được hiển thị và vận hành như nhau (các biểu tượng chẳng hạn)
Hiển thị	Trạng thái vận hành bao giờ cũng nên được hiển thị và thông tin cần thiết được cung cấp một cách thích hợp sao cho người dùng có thể có cảm giác chủ động.
Chức năng	Các chức năng trợ giúp trực tuyến, chức năng khôi phục, phím lối tắt v.v. nên được xây dựng trong màn hình với xem xét được nêu cho mức người dùng thành thạo.
Thuật ngữ, mầu sắc	Thuật ngữ có thể dễ hiểu cho người dùng nên được sử dụng. Mầu sắc nên được chọn lựa để không chọc tức hay làm mệt người dùng.

② Chuẩn bị góc nhìn chung về màn hình GUI

Cần chuẩn bị một hình ảnh biểu lộ cách bố trí toàn bộ màn hình, và mối quan hệ giữa các yếu tố trên màn hình. Các yếu tố cơ sở trên màn hình bao gồm các menu chính và menu hệ con, và các thao tác riêng cần thực hiện.

③ Tạo ra luồng màn hình GUI

Thủ tục tương tác dùng các màn hình cần được thiết kế.

<Những điểm quan trọng>

- Sau khi màn hình hiện thời biến mất, màn hình tiếp cần mở ra.
- Vị trí màn hình là cố định.
- Khi nút "đóng" được nhấn, màn hình được gọi phải được vẽ ra.
- Để dùng màn hình chỉ để nhắc người dùng đưa vào, cần dùng hộp thông báo và màn hình hiện thời vẫn còn mở.
- Thông báo cảnh báo xuất hiện để lôi kéo sự chú ý của người dùng, nếu nút "đóng" được nhấn mà không cập nhật (cất giữ) dữ liệu.

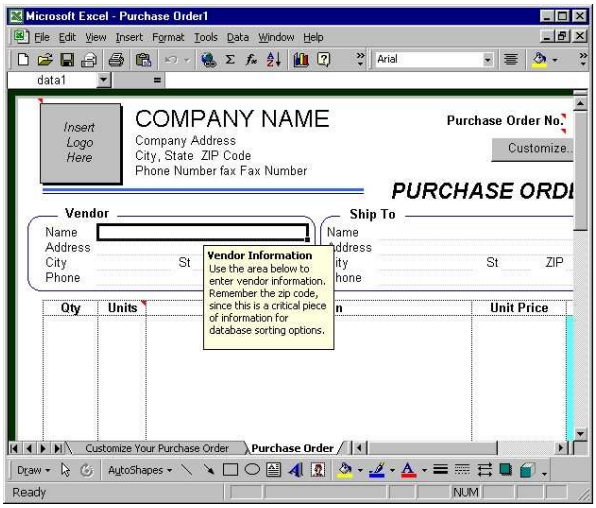
④ Xác định phương pháp đưa vào màn hình GUI

Cần xác định phương pháp đưa vào màn hình GUI.

<Những điểm quan trọng>

- Nếu người dùng là người mới bắt đầu, thì phương pháp chọn lựa như menu kéo xuống được khuyến cáo.
- Nếu người dùng có kinh nghiệm, thì phương pháp đưa vào trực tiếp (đưa thẳng văn bản vào) lại là mong muốn để tiết kiệm thời gian.
- Các khoản mục nên được bố trí theo cùng thứ tự chúng đã được bố trí trong mẫu đưa vào dữ liệu, để làm tối thiểu lỗi đưa vào.
- Các khoản mục nên được bố trí từ đỉnh tới đáy và từ trái qua phải.
- Một báo động nên được đưa ra khi một lỗi đưa vào xuất hiện, đồng thời một hộp thông báo nên xuất hiện để lôi kéo sự chú ý của người dùng.

Hình 3-4-10 Ví dụ về màn hình đưa vào



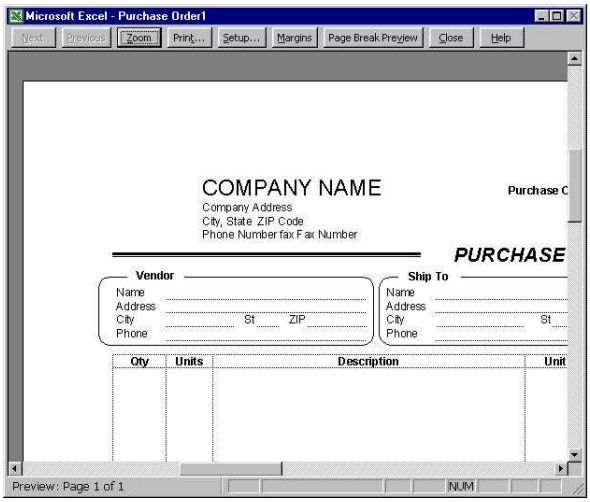
⑤ Xác định phương pháp đưa ra màn hình GUI

Trên màn hình GUI, nên có hỗ trợ cho chức năng xem trước cho phép dữ liệu được biểu lộ dưới định dạng in ra. Cũng vậy, chức năng hiển thị kết quả duyệt nên được hỗ trợ. (Xem Hình 3-4-11.)

<Những điểm quan trọng>

- Các nút "back" và "next" nên được đặt vào, vì có thể xảy ra là tất cả các dữ liệu không thể biểu thị được hết lên màn hình do giới hạn kích thước của màn hình.
- Nút "print" nên có để cho phép người dùng bắt đầu in những thứ có trên màn hình.
- Mầu sắc và khối lượng dữ liệu được hiển thị trên màn hình nên được giới hạn để cho dữ liệu được hiển thị có thể trông rõ ràng.

Hình 3-4-11 Ví dụ về màn hình đưa ra



⑥ Thiết kế bố trí màn hình GUI

Một phương pháp thiết kế cách bố trí màn hình là vẽ ra bố trí màn hình trên một mẫu đặc biệt. Phương pháp nữa là dùng ngôn ngữ trực quan và làm bản mẫu; bản mẫu được đánh giá, sửa đổi và cải tiến để hoàn chỉnh cách bố trí màn hình gần nhất với ảnh màn hình lí tưởng.

(4) Các điểm cần xem xét khi thiết kế màn hình

Các điểm sau cần được xem xét khi thiết kế màn hình:

① Chuẩn hoá màn hình

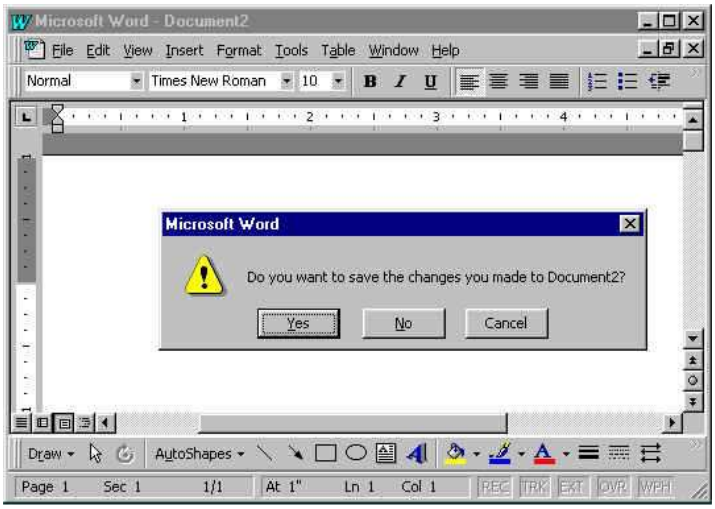
Bởi vì môi trường GUI cho phép tự do thiết kế mức cao, nên các bố trí màn hình đầy đủ là hay bị thay đổi và nhất là khi nhiều người cùng tham gia vào công việc thiết kế. Cho nên khi có nhiều người, nhiều ý kiến thì những phần cơ bản của màn hình phải được chuẩn hoá như sau:

ví dụ

- Vùng tiêu đề và vị trí đưa vào dữ liệu
  Tiêu đề và ngày tháng phải được đưa vào trong vùng tiêu đề.
- Vùng nút
  Vị trí của nút xác định chức năng tiếp cần được xác định.

- Vị trí và nội dung của thông báo (kể cả thông báo mầu sắc)
  Các yêu cầu cơ sở về hộp thông báo nên được xác định. (Chỉ nút OK hay các nút làm lại và xoá bỏ là được cần tới?)

Hình 3-4-12 Hộp thông báo



- Luồng các hình mẫu tiến trình
  Cần xác định luồng cơ sở; chẳng hạn menu chính → menu con → từng tiến trình.
- Dùng các phím PF (chức năng chương trình)
  Phải xác định xem liệu các phím PF có được dùng hay không.
- Bấm chuột
  Cần thiết lập các qui tắc chuẩn về bấm kép và bấm chuột.
- Dùng biểu dụ
  Nên dùng các biểu dụ để làm cho người dùng dễ hiểu.

Hình 3-4-13 Biểu dụ

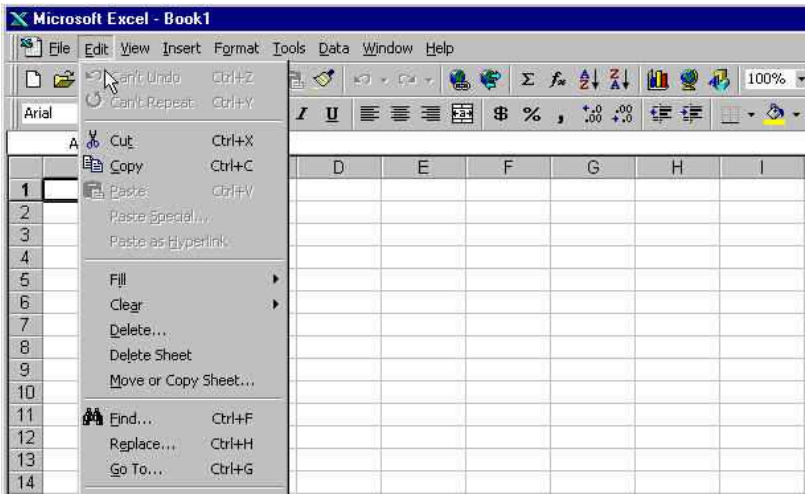
- Chuẩn hoá tiến trình tương tác
  Nên chuẩn hoá tương tác bằng việc dùng hộp thông báo v.v...
- Mầu sắc
  Số các mầu nên được giữ ở mức năm hay ít hơn. Mầu đỏ và các mầu sáng nên được dùng để lôi kéo sự chú ý vào một vùng nhỏ, trong khi mầu xanh nhạt và các mầu khác nên được dùng để biểu thị cho miền lớn.

Một định dạng chuẩn như được mô tả ở trên sẽ làm tăng tính dễ dùng.

② Phím tắt

Trong khi chuẩn bị cái nhìn toàn bộ về màn hình, đừng chỉ có hiển thị các menu dần từng bước và chọn các đặc trưng, mà còn phải cung cấp cả những đặc trưng chọn trực tiếp (các phím tắt) tương ứng với mức độ thành thạo của người dùng. Điều này làm nhẹ bớt tải công việc, và làm ngắn bớt thời gian đưa vào, trong khi làm giảm thời gian và công sức toàn thể. (Xem Hình 3-4-14.)

Hình 3-4-14 Phím tắt



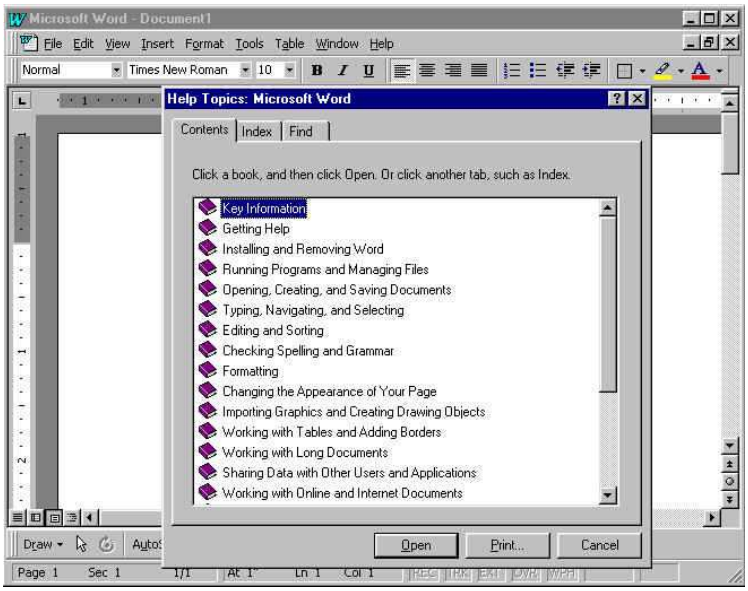
③ Hành động cần thực hiện khi màn hình bị khoá, hay phạm phải lỗi đưa vào v.v..

Màn hình phải được thiết kế để bao quát cả lối ra khẩn cấp khi một chức năng bị buộc phải kết thúc thì có thể được dùng để thoát khỏi chương trình, khi màn hình bị khoá cũng như cần có hộp thông báo đưa ra lời cảnh báo.

④ Khuôn mẫu màn hình

Bởi vì nhiều màn hình (đa cửa sổ v.v..) phải được dùng để thực hiện một nhiệm vụ, nên luồng các khuôn mẫu màn hình phải được thiết kế để cho phép người dùng xem các màn hình khác một cách có hiệu quả. Chức năng xoá luồng các khuôn mẫu kế tiếp (nút cắt bỏ) cũng như trợ giúp trực tuyến nên được hỗ trợ.

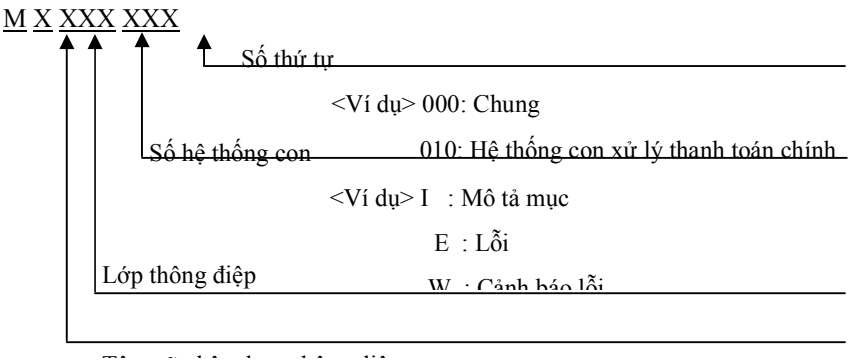
Hình 3-4-15 Chức năng trợ giúp trực tuyến



(5) Thiết kế thông báo

Nếu một thông báo về các điều kiện hệ thống được thể hiện trên màn hình, thì tính hiệu quả vận hành hệ thống sẽ tăng lên. Màn hình nên được thiết kế để hiển thị trạng thái bên trong, như lỗi in ra, việc nhấn vào phím không hợp lệ, và lỗi dữ liệu. (Các lỗi nên được thể hiện bằng mầu đỏ trong khi một báo động được đưa ra cùng lúc đó.)
Bởi vì mã dễ xử lí hơn, nên cũng phải thiết kế về mã (mã nội bộ).
Hình 3-4-16 đưa ra một ví dụ về thiết kế thông báo.

Hình 3-4-16 Thiết kế thông báo



Mã thông điệp	Thông điệp	Mô tả
ME000001	Phím nhấn là khóa chức năng không hợp lệ	Nhấn khóa chức năng sai
MG000002	Sau khi kiểm tra dữ liệu nhập vào, nhấn khóa thực hiện	Thúc đẩy người dùng kiểm tra dữ liệu
ME000001	Khóa đã tồn tại	Khóa sai được thiết lập sử dụng?
MI000002	Đặt giấy lên máy in	Giấy không đặt trong máy in
ME000003	Dữ liệu được làm nổi bật trong video đảo ngược chứa các lỗi	Nhập dữ liệu chứa lỗi

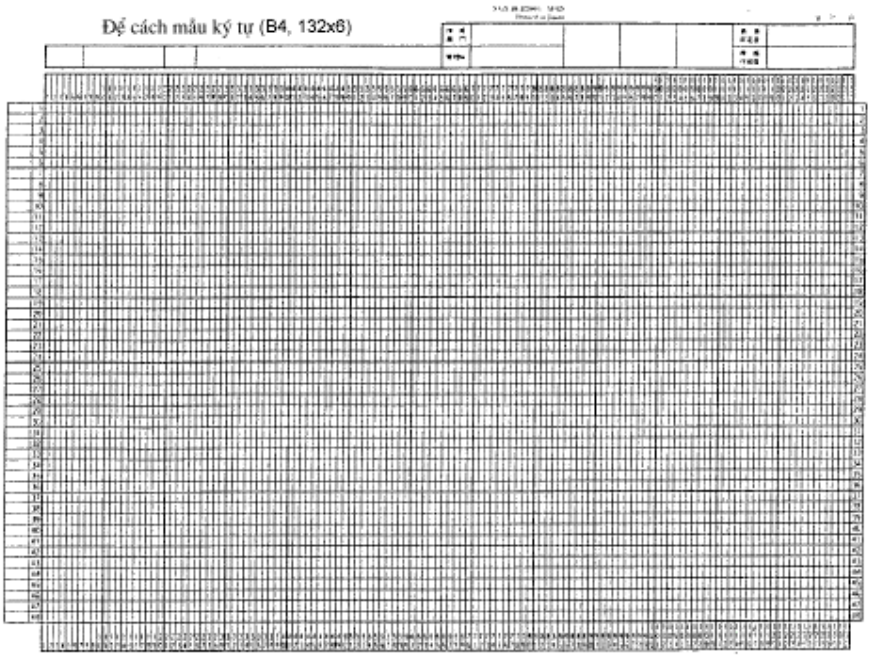
3.4.3 Thiết kế dữ liệu đưa ra chi tiết

Người dùng đưa vào dữ liệu, để cho máy tính xử lí nó, và đưa dữ liệu ra trên màn hiển thị hay trên giấy để kiểm tra kết quả của việc xử lí dữ liệu. Các mẫu đưa ra dữ liệu cần phải được thiết kế để cho nó dễ hiểu và được dùng cho các thao tác xử lí dữ liệu.

(1) Thủ tục và nhiệm vụ thiết kế báo cáo

Bởi vì dàn bài của báo cáo đưa ra đã được thiết lập trong giai đoạn thiết kế ngoài, nên khuôn mẫu đưa dữ liệu ra phải được thiết kế bằng việc dùng mẫu chuyên dụng (sơ đồ để cách) dựa trên dữ liệu do thiết kế ngoài cung cấp. (Xem Hình 3-4-17.)
Nếu một khuôn mẫu đưa dữ liệu ra được dùng, thì nó nên được thiết kế theo chi tiết nhiều hơn dựa trên dàn bài của khuôn mẫu đã được tạo ra trong giai đoạn thiết kế ngoài.

Hình 3-4-17 Sơ đồ để cách (ví dụ)



Những điểm sau nên được xét tới khi chuẩn bị thiết kế bằng việc dùng sơ đồ để cách:

- Một dòng có 132 vị trí kí tự (trong trường hợp của Hình 3-4-17).
- Một trang có 48 dòng (trong trường hợp của Hình 3-4-17).
- Về nguyên tắc, một kí tự nên được đưa vào từng ô một. Trong trường hợp kí tự chữ biểu ý thì đưa vào trong hai ô.

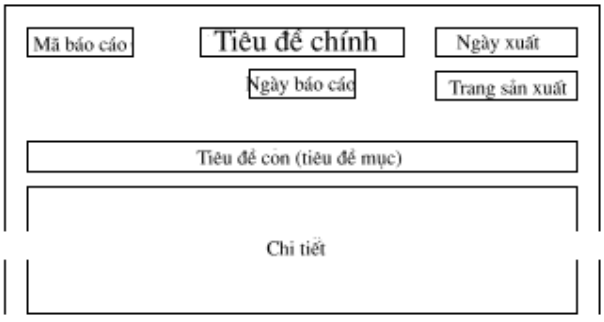
Khuôn mẫu đưa dữ liệu ra là thủ tục bao gồm ba bước được nêu dưới đây:

① Xác định vị trí tiêu đề và các dòng chi tiết

Vị trí của tiêu đề (các tiêu đề lớn, vừa và nhỏ) và đặc tả về các dòng chi tiết được xác định.

- Vị trí tiêu đề
  Vị trí của tên báo cáo đưa ra, số trang (thường được đưa vào ở góc trên bên phải), ngày tháng báo cáo được chuẩn bị (thường ở góc trên bên phải), và tiêu đề con cho các mô tả chi tiết được xác định Vị trí của tiêu đề con nói riêng phải được xác định cẩn thận vì nó ảnh hưởng tới số vị trí kí tự mà theo đó từng khoản mục dữ liệu phải được đưa vào (Xem Hình 3-4-18.)

Hình 3-4-18 Xác định vị trí của tiêu đề



- Các dòng nơi dữ liệu chi tiết được đưa vào
  Việc đặt khoảng cách giữa các dòng và vị trí của dòng cuối cùng cần được xác định. Trong Hình 3-4-17, các kí tự có thể được in ra trên 48 dòng một trang. Cách bố trí

này nên được thiết kế sao cho dữ liệu được đưa vào trên những dòng này là thấy được và dễ hiểu với người dùng.

② Xác định vị trí của các khoản mục dữ liệu và số chữ số

Các vị trí của khoản mục dữ liệu được in trên từng dòng và số các chữ số được xác định.

- Vị trí của khoản mục dữ liệu
  Các khoản mục dữ liệu của cùng kiểu được in cận kề bên nhau. Cũng vậy, người dùng có thể hình dung ra tổng số dễ dàng nếu các khoản mục dữ liệu được thu xếp từ trái sang phải.
- Xác định số các chữ số
  Số các hộp nên được xác định với số dư dự trữ nào đó. Khi tổng số được in ra, cần dự kiến để có thể in ra số lớn tối đa.
  Với các số, việc bỏ các số không đứng trước nên được thực hiện để làm cho chúng thành dễ thấy.

Hình 3-4-19 Các thể hiện số các chữ số cho từng khoản mục dữ liệu

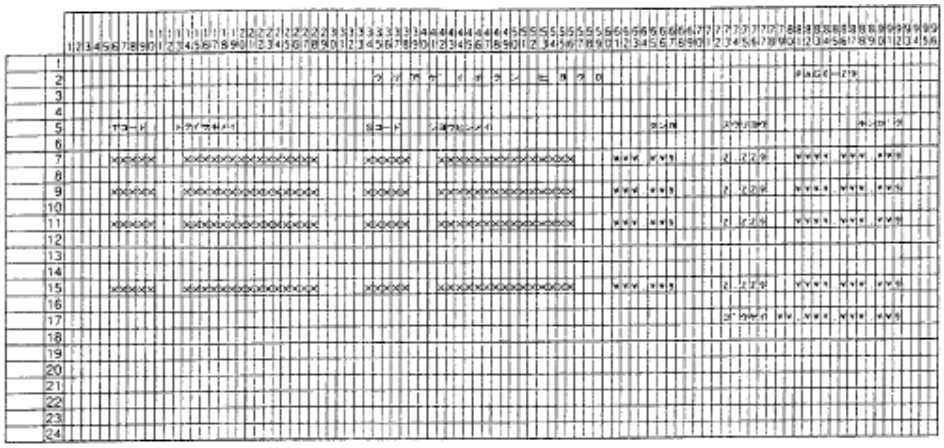
Số các chữ số có thể được soạn thảo như sau để làm cho các chữ số thành dễ thấy.

----,--9	→ -54,000
-$$$,$$$	→ -$54,000
-zzz,zz9	→ - 54,000

③ Thiết kế bố trí báo cáo

Với tất cả những xem xét trên, dữ liệu phải được đưa vào trong sơ đồ có chỗ trống để thiết kế báo cáo dữ liệu đưa ra. (Xem Hình 3-4-20.)

Hình 3-4-20 Thiết kế báo cáo đưa dữ liệu ra



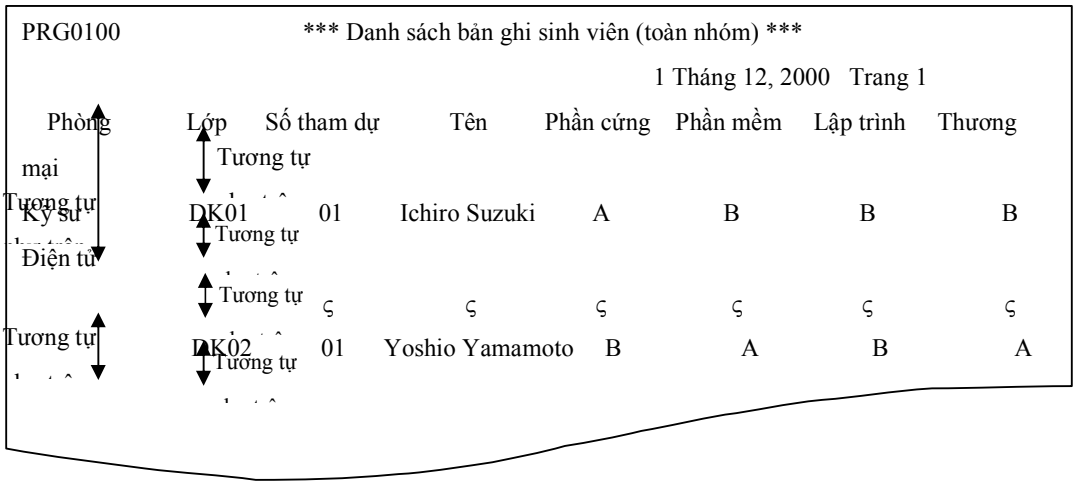
(2) Những điểm cần xem xét khi thiết kế báo cáo

Báo cáo nên được thiết kế với xem xét đầy đủ về tính dùng được và "dễ đọc." Tương tự như vậy, những điểm sau nên được xem xét khi thiết kế báo cáo:

- Vị trí tiêu đề và các khoản mục đưa ra nên được chuẩn hoá nhiều nhất có thể được.
  Nếu các vị trí tiêu đề và các khoản mục đưa ra được chuẩn hoá ở bên trong, thì báo cáo là đều nhau, dễ thấy và xử lí.

- Nên dùng những định dạng dễ thấy.
  Với những dòng có chứa dữ liệu chi tiết, nên xác định khoảng cách dòng, vị trí kí tự, việc điều chỉnh khoản mục dữ liệu (bỏ các số không v.v..), kích cỡ font v.v… để làm cho toàn bộ báo cáo thành một kết cấu dễ nhìn.
- Dữ liệu nên được thu xếp theo mức quan trọng từ trái sang phải và từ trên xuống dưới.
  Các khoản mục dữ liệu ở các vị trí bên trái nhất và bên phải nhất được nhấn mạnh.
- Các xâu kí tự nên được canh trái, và số nên được canh phải.
  Các xâu kí tự đưa ra nên được canh trái. Các vị trí không dùng nên được bỏ trống như dấu cách. Các số nên được canh phải và được hiệu chỉnh khi cần thiết.
- Các định dạng được lập theo nhóm.
  Dữ liệu của cùng kiểu nên được gắn với nhau thành nhóm, và được thu xếp trong định dạng dễ nhìn. (Xem Hình 3-4-21.)

Hình 3-4-21 Thiết kế định dạng cho từng nhóm



3.5 Tạo ra và dùng lại các bộ phận

3.5.1 Khái niệm về tạo ra và dùng lại các bộ phận

Nói chung, thách thức của việc dùng lại phần mềm là dễ 'tìm', 'đăng kí' và 'thay đổi'. Nếu các bộ phận phần mềm được xem như phương tiện cho việc dùng lại, thì những thách thức này bị thay thế bởi 'sự nhất quán của các bộ phận xem như các đơn vị', 'hệ thống hoá của các bộ phận', và 'tính độc lập của các bộ phận'.
Trong khái niệm hướng đối tượng, thì đơn vị của bộ phận phần mềm là một đối tượng. Các đối tượng được hệ thống hoá trong thư viện lớp, và tính độc lập của đối tượng được thực hiện qua việc bao bọc. Do đó, khái niệm hướng đối tượng có thể được nói là thích hợp cho việc dùng lại các sản phẩm công việc.

3.5.2 Dùng gói phần mềm

(1) Thư viện chương trình con

Thư viện chương trình con từ lâu đã được biết tới như gói chương trình con khoa học - scientific subroutine package (SSP), trong lĩnh vực tính toán khoa học và kĩ nghệ. Các chương trình con và hàm phức tạp không do người dùng phát triển mà được các nhà chế tạo, các trường đại học và viện nghiên cứu tổng hợp thành thư viện sẵn có cho người dùng.
Khái niệm này đã được áp dụng cho việc phát triển các chương trình ứng dụng nghiệp vụ, và các chương trình con có ích được cung cấp như các gói cho người dùng thông thường, không mất tiền. Bằng cách dùng các gói này, người ta đạt tới việc rút gọn đáng kể giờ công cần để phát triển phần mềm. Tuy nhiên hệ thống ứng dụng nghiệp vụ lại chứa các vật phẩm đã được phân loại hay tri thức sở hữu riêng. Do đó, một công ti tổ chức các chương trình con được phát triển dựa trên các chuẩn nội bộ, thành thư viện và cho phép chúng được dùng lại chỉ trong nội bộ.

(2) Thư viện lớp

Trong phát triển hướng đối tượng, các đối tượng được bao bọc để những thay đổi của một đối tượng này không ảnh hưởng tới các đối tượng bao quanh. Có thể duy trì tính độc lập của từng đối tượng. Do đó, các đối tượng có thể được dùng lại như các bộ phận dễ dàng hơn các bộ phận trong chương trình được phát triển bằng việc dùng ngôn ngữ thủ tục. Chẳng hạn trong trường hợp ngôn ngữ Java, các đối tượng thường được dùng được tập hợp trong thư viện lớp. Bằng việc dùng thư viện lớp, thời gian và lao động cần cho phát triển chương trình có thể được tiết kiệm.

3.6 Tạo ra tài liệu thiết kế trong

Tài liệu thiết kế trong được tạo ra dựa trên dữ liệu do thiết kế trong cung cấp. Thiết kế trong được chuẩn bị dựa trên tài liệu thiết kế ngoài theo quan điểm của người phát triển. Bởi vì tài liệu thiết kế trong được người phát triển duyệt xét, nên việc dùng các thuật ngữ kĩ thuật trong tài liệu này là được phép. Cũng vậy, tất cả các chi tiết, kể cả hệ con, chương trình và ngoại lệ, đều phải được mô tả trong tài liệu thiết kế trong.
Kết quả của công việc thiết kế hệ thống đặc biệt được biên soạn thành tài liệu thiết kế trong.
Thiết kế trong là một tiến trình thiết kế quan trọng vì nó có trách nhiệm thoả mãn mọi yêu cầu người dùng do thiết kế ngoài xác định. Bất kì sai lầm thiết kế nào cũng đều có thể có tác động nghiêm trọng lên các tiến trình về sau. Phải luôn nhớ rằng một phần nhỏ của hệ thống được thiết kế trong giai đoạn thiết kế hệ thống và do đó người ta phải nhớ rằng bất kì thay đổi nào trong thiết kế đã hoàn chỉnh đều không thể dễ dàng được thực hiện.
Tài liệu thiết kế trong bao gồm các phần tử được vẽ trong Hình 3-6-1.

Hình 3-6-1 Các yếu tố trong tài liệu thiết kế trong

3.6.1 Tổ chức tài liệu thiết kế trong

(1) Chính sách thiết kế trong

Trong chính sách thiết kế trong, cần mô tả một tổng quan về chính sách phát triển hệ thống trước khi bắt đầu công việc thiết kế.

① Phương pháp thiết kế

Mô tả cho phương pháp thiết kế được chọn cho công việc thiết kế trong. Thông thường phương pháp thiết kế có cấu trúc được sử dụng. Mô tả cách dùng phương pháp thiết kế có cấu trúc được xác định với việc nói tới các phương pháp thiết kế đã được xét duyệt khác.

② Kĩ thuật làm tài liệu

Các kĩ thuật làm tài liệu như: HIPO, DFD, lưu đồ, sơ đồ bọt v.v... Mô tả cho các kĩ thuật được chấp thuận và cách chúng được chấp nhận.

③ Nhiệm vụ thiết kế

④ Các công việc khác

Mô tả thủ tục ghi lại thay đổi, chi tiết về xét duyệt thiết kế và các vấn đề khác phải xác định trước khi bắt đầu công việc thiết kế.

(2) Cấu hình hệ thống

Trong cấu hình hệ thống, mô tả cho các chi tiết về hệ thống và hệ con được nêu sau đây.

① Tổng quan hệ thống (luồng hệ thống)

Trong tổng quan hệ thống, mô tả cho luồng hệ thống mới (DFD hệ thống mới) được tạo ra bởi phân tích hệ thống. Nếu thay đổi được thực hiện trong luồng hệ thống mới do việc cấu trúc hay phân hoạch chức năng trong thiết kế hệ thống thì cần mô tả cho luồng hệ thống mới đã thay đổi.

② Sơ đồ cấu trúc hệ thống (biểu đồ nội dung)

Biểu đồ các nội dung được đưa ra trong Hình 3-2-13.

③ Giao diện giữa các chương trình

Biểu đồ nêu ra cái vào cho tới cái ra từ một chương trình được vẽ ra. Sử dụng biểu đồ luồng dữ liệu (DFD), lưu đồ v.v... (Xem Hình 3-2-14.)

④ Biểu đồ quan hệ chương trình

Thường đưa ra luồng tiến trình biểu thị cho trình tự thực hiện của chương trình.

⑤ Bản in chương trình

Cần chuẩn bị bản in chương trình chỉ ra tên của chương trình được dùng trong hệ thống, tổng quan vận hành v.v...

(3) Chức năng chương trình

Trong các chức năng chương trình có mô tả về các chương trình đã được phân chia bằng việc dùng biểu đồ cái vào xử lí cái ra (IPO).

① Biểu đồ chung

Mô tả một tổng quan về các hệ con và chương trình.

② Biểu đồ chi tiết

Mô tả các chi tiết của từng đơn vị chức năng riêng (chương trình).

(4) Bố trí màn hình

Trong bố trí màn hình, các tài liệu sau được nêu ra:
- Cái nhìn chung về màn hình
- Biểu đồ chuyển màn hình

(5) Bố trí báo cáo vào/ra

Danh sách các báo cáo được dùng trong hệ thống, các giải thích chi tiết về từng báo cáo và

việc bố trí của cả hai loại tài liệu gốc đưa vào và báo cáo đưa ra được đính kèm. (Xem Hình 3-4-2 và 3-4-20.)

(6) Cấu hình tệp

Danh sách các tệp được dùng trong hệ thống, những giải thích chi tiết về từng tệp và cách bố trí tệp được gắn với cấu hình tệp. (Xem Hình 3-3-4.)
Tên tệp, dung lượng tệp, tên thư viện và vị trí (số hiệu đĩa) được mô tả trong danh sách các tệp. Tương tự như vậy, đặc tả cho các bảng được dùng trong hệ thống cũng được đính kèm.

(7) Hiệu năng hệ thống

Hiệu năng hệ thống được tính toán và đánh giá dựa trên hiệu năng máy tính, dung lượng bộ nhớ và hiệu năng của thiết bị nhớ, bản thân phần mềm hệ thống và xử lí, môi trường hệ thống v.v. Mô tả cho hiệu năng hệ thống đã được tính toán.

<Tính hiệu năng hệ thống >

- Khối lượng trung bình các dữ liệu được sinh ra và khối lượng dữ liệu trong giờ cao điểm
- Thời gian xử lí dữ liệu (cho một khoản mục hay trong một thời kì xác định)

(8) Kế hoạch kiểm thử

Mô tả kế hoạch kiểm thử tích hợp.
- Khối lượng dữ liệu

3.6.2 Các điểm cần lưu ý khi tạo ra tài liệu thiết kế trong

Các điểm sau đây cần được lưu ý khi tạo ra tài liệu thiết kế hệ thống:

- Tài liệu thiết kế trong phải bao hàm tất cả các chức năng được mô tả trong tài liệu thiết kế ngoài.
  Mặc dầu thiết kế trong được tạo nên theo quan điểm của người phát triển, nó vẫn phải dựa trên thiết kế ngoài được tạo ra theo quan điểm của người dùng. Bởi vì nhân tố hàng đầu trong việc phát triển hệ thống là người dùng, cho nên cần xác nhận lại rằng tất cả mọi yêu cầu của người dùng đều được bao hàm trong tài liệu thiết kế trong.
- Tất cả các chương trình đều phải được mô tả rõ ràng.
  Không chỉ mô tả những phần chính của chương trình mà còn cả các phần ngoại lệ, sửa chữa, sao lưu và các yêu cầu chương trình khác.
- Tất cả các tệp và tất cả cái vào và cái ra đều phải được mô tả rõ ràng.
  Tất cả các tệp, màn hình và báo cáo vào/ra được dùng trong hệ thống phải được mô tả rõ ràng.
- Mọi thủ tục giải quyết lỗi đều phải được mô tả rõ ràng.
  Cách thức lỗi được xử lí ảnh hưởng tới độ tin cậy hệ thống. Tương tự như vậy, phải mô tả rõ thời gian giải quyết lỗi.
- Phải tôn trọng chuẩn làm tài liệu, cần tránh các cách diễn đạt sai lầm.
  Văn bản trong tài liệu phải được viết theo cách thức rõ ràng, chính xác.

3.6.3 Kiểm điểm thiết kế

Việc kiểm điểm được tiến hành trong giai đoạn thiết kế của tiến trình phát triển phần mềm được gọi là kiểm điểm thiết kế. Nó bao gồm các đặc tả thiết kế. Bởi vì chất lượng sản phẩm được xác định theo đặc tả thiết kế, nên cuộc họp kiểm điểm là rất quan trọng để cải tiến chất lượng phần mềm.

(1) Phương pháp kiểm điểm

Kiểm điểm thiết kế chủ yếu được tiến hành vào tiến trình thiết kế hệ thống và thiết kế chương trình. Các tài liệu là chủ đề cho cuộc họp kiểm điểm thiết kế bao gồm những tài liệu sau:

- Tài liệu thiết kế trong
- Biểu đồ quan hệ chương trình
- Các hướng dẫn về chức năng chương trình
- Tài liệu thiết kế màn hình chi tiết
- Tài liệu thiết kế báo cáo chi tiết
- Tài liệu thiết kế tệp chi tiết
- Tài liệu thiết kế dữ liệu kiểm thử

Những chuẩn bị sau phải được thực hiện trước khi bắt đầu kiểm điểm thiết kế:

- Lập kế hoạch kiểm điểm thiết kế
  Phải điều chỉnh lịch và đệ trình cho tổ chức chịu trách nhiệm thực hiện kiểm điểm thiết kế.
- Phân phối tài liệu có liên quan trước khi bắt đầu kiểm điểm thiết kế
  Những tài liệu đã chuẩn bị và các tài liệu có liên quan được phân phát cho những người tham dự vào kiểm điểm thiết kế, để cho họ có thể kiểm tra chúng trước khi kiểm điểm cụ thể vào thiết kế
- Chuẩn bị danh sách kiểm điểm
  Bằng việc đối chiếu vào danh sách kiểm điểm, cần kiểm kĩ lưỡng tính thích hợp của nội dung các khoản mục, tính nhất quán, việc bỏ sót, sự tuân thủ với chuẩn nội bộ, sự rõ ràng, tính hiểu được và các điểm khác. Danh sách kiểm điểm phải bao gồm tất cả những điểm này để cho sự vật được kiểm điểm có thể được đánh giá đúng.

Trong kiểm điểm thiết kế thực tế, các tài liệu được tạo ra trong giai đoạn thiết kế trong được sánh với những đặc tả thiết kế được tạo ra giai đoạn thiết kế ngoài. Chất lượng của từng tài liệu xác định được đánh giá, và mức độ tuân thủ các đặc trưng chất lượng được yêu cầu sẽ được kiểm điểm.
Thời gian dành cho việc kiểm điểm thiết kế là xấp xỉ hai giờ cho mỗi kiểm điểm, mà cũng có thể khác, tuỳ theo kích cỡ của chương trình. Việc kiểm điểm thiết kế này nên được thực hiện thông thường một hay hai lần.

(2) Hệ thống kiểm điểm

Những người được nêu dưới đây đóng vai trò trung tâm trong việc thực hiện kiểm điểm thiết kế:

- Những người thiết kế có cùng mức kĩ năng kĩ thuật như những người trực tiếp chịu trách nhiệm cho việc tạo ra chương trình trong giai đoạn thiết kế trong
- Những nhân viên liên quan tới tiến trình thiết kế

Những người cấp cao hơn người thiết kế và những nhân viên liên quan không nên tham gia vào cuộc kiểm điểm thiết kế. Điều này là vì họp kiểm điểm thiết kế nhằm đánh giá các tài liệu,

không đánh giá khả năng của người thực hiện kiểm điểm thiết kế.
Nói chung, một bộ phận chuyên cho kiểm điểm thiết kế được thành lập độc lập với tổ chức mà những người thiết kế và những nhân viên liên quan này thuộc vào. Do đó, kiểm điểm thiết kế có thể được tiến hành theo quan điểm của bên thứ ba.

(3) Sự tham dự của người dùng

Thiết kế màn hình chi tiết hay thiết kế báo cáo chi tiết, là công việc tạo ra giao diện giữa hệ thống và người dùng. Do đó, tổ chức người dùng có thể tham gia vào kiểm điểm thiết kế khi có nhu cầu phát sinh.

Bài tập

Q1 **Công việc nào là công việc thích hợp nhất được làm ở thiết kế trong, xem như một phần của hoạt động phát triển hệ thống?**

a. Thiết kế mã
b. Thiết kế dữ liệu vật lí
c. Thiết kế cấu trúc chương trình
d. Xác định yêu cầu
e. Thiết kế dữ liệu vật lí

Q2 **Hãy <u>chọn hai</u> nhiệm vụ cần được thực hiện trong thiết kế dữ liệu vật lí trong giai đoạn thiết kế trong.**

a. Ước lượng thời gian truy nhập và dung lượng
b. Xác định khoản mục dữ liệu c. Phân tích mối quan hệ dữ liệu
d. Tạo ra đặc tả tệp
e. Xác định cách bố trí bản ghi

Q3 **Kĩ thuật nào là kĩ thuật để biểu diễn các chức năng và luồng dữ liệu bằng các kí hiệu chỉ ra luồng dữ liệu, xử lí (chức năng), kho dữ liệu và nguồn ngoài (nguồn dữ liệu được sinh ra và gửi đi)? (Kĩ thuật này là một trong các phương pháp phân tích có cấu trúc.)**

a. DFD b. ERD c. Sơ đồ NS d. Biểu đồ chuyển trạng thái e. biểu đồ Warnier

Q4 **Mô tả nào là mô tả thích hợp về HIPO, một phương pháp thiết kế có cấu trúc?**

a. Biểu đồ các nội dung và biểu đồ luồng dữ liệu được dùng.
b. Thông tin điều khiển được truyền qua giữa các khối xử lí được mô tả cùng với các mũi tên trong biểu đồ nội dung.
c. Biểu đồ nội dung chỉ ra chức năng toàn thể của chương trình, và các số được đưa vào trong các khối xử lí chỉ ra trình tự xử lí.
d. Các kí hiệu trong lưu đồ được dùng để chỉ ra cái gì được chọn và cái gì được lặp lại.
e. Mối quan hệ giữa các bước vào/ra và xử lí có thể được biểu diễn rõ ràng.

Q5 **Trong xem xét được nêu dưới đây về thiết kế màn hình trong các giai đoạn thiết kế ngoài và trong, xem xét nào là xem xét <u>không thích hợp</u>?**

a. Việc chuyển màn hình nên được thiết kế với việc xem xét không chỉ việc lựa từng bước bằng việc dùng một menu, mà còn với truy nhập trực tiếp vào màn hình mong muốn cho người dùng đã thành thạo.
b. Từng khoản mục mà dữ liệu được đưa vào trên màn hình phải được bao trong ngoặc vuông, để nhấn mạnh rằng nó là trường đưa vào dữ liệu.
c. Cách bố trí màn hình phải được thiết kế theo cách các khoản mục cần tham chiếu được thu xếp từ trái sang phải, và từ trên xuống dưới.
d. Để hoàn thành tiến trình đang diễn ra, màn hình phải được thiết kế để ngăn cản người dùng khỏi bỏ việc đưa vào dữ liệu, hay trở lại màn hình trước đó.
e. Bố trí màn hình phải được chuẩn hoá; các qui tắc về vị trí cho hiển thị tiêu đề và thông báo phải được thiết lập.

4 Thiết kế chương trình

Mục đích của chương

Thiết kế chương trình là một tiến trình quan trọng vì nó làm cho các nhiệm vụ lập trình được trôi chảy.
Nếu một chương trình được phát triển với giao diện rõ ràng và các mô đun được thiết kế như những hộp đen, thì các bộ phận của chương trình này có thể được dùng lại hay chúng có thể được dùng để tạo ra chương trình mới. Có thể có khả năng xây dựng hệ thống chất lượng cao bằng việc tổ hợp đơn giản các mô đun chất lượng cao.
Chương này mô tả từng nhiệm vụ thiết kế chương trình cũng như các sản phẩm công việc trong giai đoạn thiết kế chương trình.

① Hiểu mục đích, những điểm quan trọng và nhiệm vụ liên quan tới thiết kế chương trình
② Hiểu nội dung và ý nghĩa của từng tiến trình trong giai đoạn thiết kế chương trình
③ Hiểu việc phân hoạch mô đun và cách đánh giá mô đun

Giới thiệu

Thiết kế chương trình là tiến trình cuối cùng trong toàn thể giai đoạn thiết kế.
Đồng thời hay song song với việc lập trình, tài liệu thiết kế chương trình được chuẩn bị trong quá khứ. Tuy nhiên, với qui mô ngày càng tăng của việc phát triển hệ thống, nhu cầu tăng năng suất và sử dụng các bộ phận, việc thiết kế chương trình không còn là công việc phụ cho lập trình nữa, bây giờ nó được coi như một trong những tiến trình phát triển hệ thống.
Trong giai đoạn thiết kế trong, hệ con được phân chia thành các đơn vị chương trình chức năng. Trong giai đoạn thiết kế chương trình, công việc chính là phân hoạch mô đun. Từng đơn vị chương trình chức năng được xác định bởi thiết kế trong, được phân hoạch ra bằng việc dùng phương pháp thiết kế có cấu trúc theo các mô đun, những đơn vị nhỏ nhất có thể được soạn thảo ra. Với việc phân hoạch chương trình thành các mô đun, một chương trình có thể được làm cho dễ hiểu hơn và dễ bảo trì hơn. Hơn nữa, để làm cho những mô đun đó là một phần của chương trình mới phát triển trong tương lai, cần có việc phân hoạch logic.
Trong giai đoạn thiết kế chương trình, nội dung của tài liệu thiết kế trong trước hết phải được hiểu một cách thấu đáo, và sau đó mục đích và thủ tục phải được thiết lập để làm những nhiệm vụ thiết kế chương trình có hiệu quả.

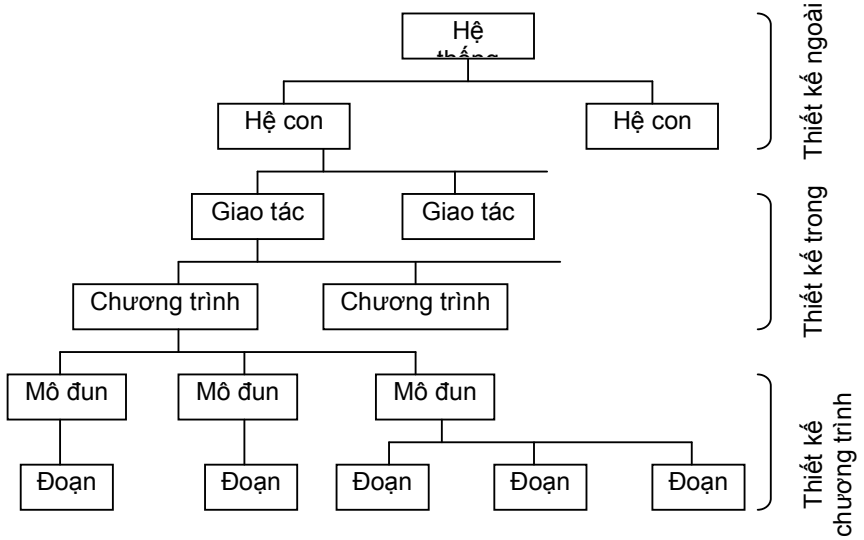
4.1 Mục đích và nhiệm vụ của thiết kế chương trình

Thiết kế chương trình là pha thứ tư theo mô hình thác đổ. Nó là pha mà trong đó các chương trình được thiết kế dựa trên tài liệu thiết kế trong.

4.1.1 Mục đích của thiết kế chương trình

Mục đích của thiết kế chương trình là thiết kế cấu trúc bên trong của chương trình. Trong thiết kế chương trình, phương pháp thiết kế có cấu trúc được dùng để phân hoạch chương trình thành các mô đun. Bằng việc phân hoạch chương trình thành mô đun và làm sáng tỏ mối quan hệ giữa các mô đun, cấu trúc của chương trình và việc bảo trì có thể được làm dễ dàng hơn.
Hình 4-1-1 đưa ra các đối tượng được giải quyết trong việc phát triển hệ thống, kể cả những đối tượng được bao hàm trong các pha trước.
Các đối tượng được giải quyết trong việc phát triển hệ thống được chia nhỏ ra tuỳ theo qui mô của việc phát triển như sau:
Hệ thống-hệ con-giao tác/công việc-chương trình-mô đun-đoạn (chức năng)-chỉ lệnh

Hình 4-1-1 Các đơn vị công việc trong phát triển hệ thống



Các mô đun bao gồm:
- Đơn vị dịch trong ngôn ngữ cấp cao
- Các khoản mục chức năng của chương trình
- Khoản mục đơn vị menu
- Chương trình gốc có từ 10 tới 300 câu lệnh
- Các nhiệm vụ (tiến trình) xử trí dưới việc quản lí nhiệm vụ (tiến trình)
- Đơn vị mô đun nạp
- Sự vật được dùng trong việc phát triển hướng sự vật
- Đơn vị biến cố giao diện người dùng đồ hoạ (GUI)

Một mô đun có thể được xác định như một đơn vị logic cố hữu về mặt định lượng hay logic. Định nghĩa này nên được hiểu chỉ như một tiêu chí.

4.1.2 Nhiệm vụ thiết kế chương trình

Công việc thiết kế chương trình được thực hiện theo các bước được nêu sau đây:
1. Xác nhận nội dung của tài liệu thiết kế trong
2. Phân hoạch thành mô đun
3. Chuẩn bị đặc tả mô đun
4. Chuẩn bị tài liệu thiết kế chương trình
5. Chuẩn bị đặc tả kiểm thử
6. Thực hiện kiểm điểm thiết kế

(1) Xác nhận nội dung của tài liệu thiết kế trong

Trong khi tiến hành thiết kế hệ thống, tính nhất quán của tất cả công việc thiết kế, kể cả thiết kế cơ sở, thiết kế ngoài, thiết kế trong và thiết kế chương trình, đều phải được duy trì. Để phản ánh nội dung của thiết kế trong như một tổng thể trong thiết kế chương trình, những điểm được nêu dưới đây phải được xem xét trước khi chương trình được xác định trong thiết

kế trong thành các mô đun.
- Xác định chức năng (phải làm gì)
- Thông tin vào (đưa vào cái gì)
- Xử lí (kiểu xử lí nào cần thực hiện)
- Cái ra (đưa ra cái gì)

(2) Phân hoạch mô đun

Việc phân hoạch mô đun là công việc phân hoạch các chức năng chương trình thành các đơn vị biên dịch, bằng cách dùng phương pháp thiết kế có cấu trúc. Nó là thiết kế chương trình lõi. Lấy chiếc xe làm ví dụ để giải thích cho việc phân hoạch mô đun.

Hình 4-1-2 Phân hoạch xe

Xe được lắp ráp với một thân xe, động cơ, bánh xe, chỗ ngồi v.v.. Nếu tài liệu thiết kế chu đáo và đặc tả về xe và các bộ phận đều có sẵn, thì việc sản xuất nó được thực hiện dễ dàng bằng việc tạo ra từng bộ phận trước khi lắp ráp các bộ phận đó lại. Nếu phanh hỏng, bộ phận của nó có thể được sửa lại hay thay thế.

Điều này áp dụng cho việc phát triển hệ thống. Không giống như trường hợp lắp ráp xe, các bộ phận trong việc phát triển hệ thống là không thấy được, do đó việc phân hoạch được thực hiện từ khía cạnh vật lí cho tới logic. Nên tránh việc chỉ dùng kích cỡ như cách đo việc phân hoạch vì nó làm phát sinh các mô đun thiếu sự thống nhất, làm khó cho việc thực hiện các nhiệm vụ lập trình hay bảo trì. Do đó, xem như một nguyên tắc mô đun phải được phân hoạch thành các đơn vị logic, và một cách đo vật lí phải được dùng làm phương tiện phụ.

Việc phân hoạch mô đun có các ưu điểm sau:

- Tính độc lập của mô đun có thể được đảm bảo.
  Bằng việc phân hoạch nội dung của xử lí được xác định trong thiết kế trong thành các đơn vị cố kết về mặt logic thay vì thành các đơn vị vật lí, tính độc lập mô đun có thể được đảm bảo.
- Hiệu quả xử lí được cải thiện
  Bằng việc tối thiểu hoá các quan hệ với các mô đun khác, hiệu quả xử lí có thể cải thiện.
- Tạo ra và dùng lại các bộ phận
  Các mô đun có thể được dùng bởi chương trình, hay các mô đun có thể được dùng lại thì đều có thể được trích ra để tạo ra các bộ phận mới, hay như các bộ phận trong các chương trình khác.
- Tính hiệu quả và độ tin cậy bảo trì được cải tiến
  Nếu cần thay đổi các chức năng hệ thống, thì chỉ những mô đun được liên kết với những thay đổi hệ thống mới có thể bị thay đổi hay thay thế, cho nên hiệu quả bảo trì và độ tin cậy có thể được cải tiến.

Thủ tục phân hoạch mô đun và các chi tiết về thiết kế cấu trúc sẽ được mô tả về sau.

(3) Chuẩn bị đặc tả mô đun

Cần xác định nội dung của việc xử lí được thực hiện bởi từng mô đun. Đặc tả mô đun phải được chuẩn bị có chú ý tới các chi tiết, không bỏ sót các chức năng.

(4) Chuẩn bị tài liệu thiết kế chương trình

Kết quả của công việc được thực hiện theo các bước (1), (2) và (3) trên được soạn thành tài liệu thiết kế chương trình, dùng làm hướng dẫn cho việc mã hoá. Trong tài liệu thiết kế chương trình, những thông tin sau đây phải được mô tả:
<Nội dung của đặc tả thiết kế chương trình >

- Đường lối thiết kế chương trình
- Tổng quan về chương trình
- Biểu đồ cấu trúc chương trình
- Chi tiết xử lí
- Trường hợp kiểm thử
- Mô tả khoản mục

(5) Chuẩn bị đặc tả kiểm thử

Đặc tả kiểm thử được chuẩn bị tương ứng theo mục đích của từng kiểm thử. Kiểm thử chương trình được phân loại thành các kiểm thử đơn vị và kiểm thử tích hợp.

(6) Kiểm điểm thiết kế

Mục đích của kiểm điểm thiết kế là:

- Làm hợp lệ việc thoả mãn yêu cầu người dùng
- Kiểm chứng sự nhất quán với thiết kế trong, và sẵn sàng cho việc chuyển sang công việc lập trình

Lưu ý tới những mục đích này, cần phải tiến hành kiểm điểm thiết kế. Các cuộc họp kiểm điểm là quan trọng ở tất cả các giai đoạn. Kết quả của việc kiểm điểm thiết kế được tiến hành trên nội dung của thiết kế chương trình có ảnh hưởng lớn tới công việc lập trình.

<Những điểm quan trọng cần xét tới khi tiến hành kiểm điểm thiết kế >

- Nội dung của tài liệu thiết kế chương trình phải được kiểm tra.
- Nội dung của tài liệu thiết chương trình phải được so sánh với nội dung của tài liệu thiết kế trong. Cần chỉ ra các chức năng thiếu và các khiếm khuyết.
- Xác nhận rằng không có các chức năng có liên quan tới mô đun mà bị thiếu.
- Xác nhận rằng việc phân hoạch mô đun đã được làm đúng
- Tính nhất quán của giao diện giữa các mô đun phải được kiểm chứng. Cũng vậy, phải kiểm tra để xác nhận rằng không có giao diện nào bị thiếu.

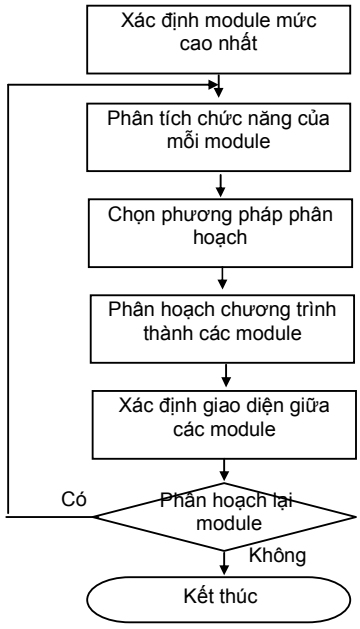
4.2 Thiết kế có cấu trúc cho chương trình

Khi qui mô hệ thống mở rộng lớn dần và các chức năng mà người dùng yêu cầu trở nên phức tạp hơn, thì chương trình được dùng để vận hành hệ thống như vậy cũng trở thành phức tạp. Kết quả là, số các lỗi tiềm tàng tăng lên, và thời gian và chi phí phải tiêu để sửa chúng cũng tăng lên.
Trong quá khứ khi phần cứng còn đắt, các chương trình đã được thiết kế bằng cấu trúc logic phức tạp để cụ thể hoá các chức năng nâng cao; và việc viết những chương trình như vậy là cơ hội cho người lập trình biểu lộ tài năng của mình. Tài nguyên hệ thống bây giờ đã sẵn có dư thừa; điều quan trọng là tạo ra chương trình chất lượng cao, dễ hiểu.

4.2.1 Thủ tục thiết kế có cấu trúc

Mục này mô tả các nhiệm vụ thiết kế có cấu trúc của việc phân hoạch một chương trình thành các mô đun. Điểm mấu chốt trong thiết kế có cấu trúc là phân hoạch chương trình theo cách nó làm tăng sự độc lập của các mô đun .

Hình 4-2-1 Thủ tục thiết kế có cấu trúc



Chi tiết về từng bước là như sau:

(1) Xác định mô đun mức cao nhất

Mô đun mức cao nhất là mô đun được gọi tới lần đầu tiên khi chương trình bắt đầu. Mô đun này thực hiện hai chức năng:

- Kiểm soát toàn bộ chương trình (từng mô đun)

- Đặt các giá trị khởi đầu cho các khoản mục dữ liệu (như bộ đếm)
- Mở và đóng tệp

Có trường hợp mà mô đun mức cao nhất chỉ kiểm soát toàn bộ chương trình, còn các mô đun khác (các mô đun mức thấp) đặt giá trị khởi đầu cho các khoản mục dữ liệu và mở/đóng tệp.

(2) Phân tích chức năng của từng mô đun

Tất cả các chức năng cốt yếu cần cho việc chạy chương trình đều được nhận diện. Chúng được phân hoạch thêm nữa ra hay được tổ hợp lại khi cần để cho chúng có thể được phân hoạch thành một tập tối ưu các mô đun.

<Các chức năng của mô đun >

- Đọc tệp
- Kiểm tra lỗi trong dữ liệu đưa vào
- Xử lí (tính toán) dữ liệu
- Đưa dữ liệu ra
- Giải quyết lỗi

(3) Chọn phương pháp phân hoạch

Với quan điểm được nói tới dưới đây, phương pháp phân hoạch mô đun thích hợp nhất cần phải được chọn ra. Từng phương pháp phân hoạch được mô tả chi tiết trong Mục 4.2.2.

- Phương pháp phân hoạch được thiết kế với chú ý tập trung vào luồng dữ liệu
  Phương pháp này là thích hợp cho hệ thống giao tác trực tuyến.
  <Bất kì một trong những phương pháp sau đây đều có thể được dùng:>
  - Phương pháp phân hoạch STS
  - Phương pháp phân hoạch TR
  - Phương pháp phân hoạch hàm chung
- Phương pháp phân hoạch được thiết kế với chú ý tập trung vào cấu trúc dữ liệu
  Phương pháp này là thích hợp cho kiểu xử lí theo lô trong đó chủ yếu xử lí các tệp.
  <Một trong hai phương pháp sau có thể được dùng:>
  - Phương pháp Jackson
  - Phương pháp Warnier

(4) Phân hoạch chương trình thành các mô đun

Bằng việc dùng bất kì phương pháp phân hoạch nào đã được nêu ở mục (3) trên, chương trình được phân hoạch thành các mô đun dựa trên các chuẩn cho việc phân hoạch mô đun. Mô đun được gọi tới được gọi là mô đun cấp dưới .

<Thủ tục phân hoạch >

1. Xác định mô đun mức cao nhất
2. Xác định các mô đun được gọi bởi mô đun mức cao nhất
3. Xác định các mô đun được gọi bởi bước 2 trên đây
4. Các mô đun được chia ra dần từng bước thành các mô đun mức thấp.

Trong việc phân hoạch, hướng dẫn sau nên được tuân thủ:

<Hướng dẫn về phân hoạch mô đun:>

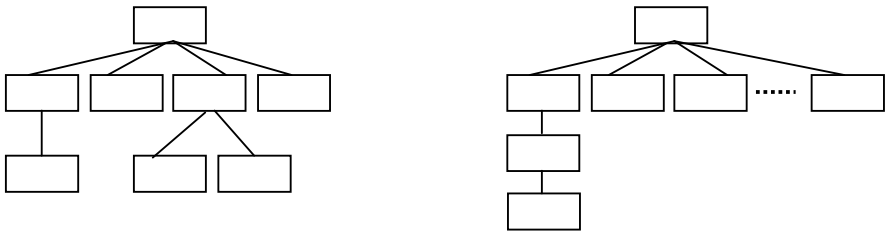
- Nếu một chương trình bao gồm từ 10 tới 300 mô đun, thì việc phân hoạch có thể được thực hiện dễ dàng.

- Số các mô đun được chứa ở một mức (phân cấp bậc) nên là mười hay ít hơn.
- Chiều sâu nên là bốn mức hay ít hơn.

Hình 4-2-2 đưa ra một trường hợp trong đó chương trình được phân hoạch đúng, và trường hợp khác nó được phân hoạch không đúng.

Hình 4-2-2 Hai trường hợp phân hoạch mô đun

<Trường hợp chương trình phân hoạch thích hợp> <Trường hợp chương trình phân hoạch không thích hợp>

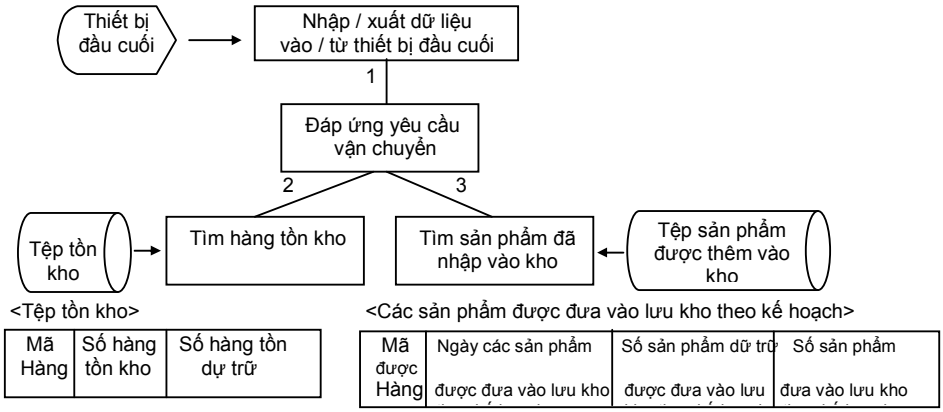


(5) Xác định giao diện giữa các mô đun

Bây giờ chúng ta mô tả dữ liệu, và các điều kiện cần để truyền dữ liệu giữa các mô đun. (Xem Hình 4-2-3.)

Việc xác định giao diện giữa các mô đun cũng quan trọng như việc phân hoạch mô đun. Thậm chí sau khi chương trình đã được phân hoạch thành các mô đun, thì độ tin cậy của bản thân chương trình cũng chỉ có thể được đảm bảo nếu từng giao diện mô đun có thể thực hiện chức năng của nó; chẳng hạn, nếu A được đưa vào, thì B được cho lại mà không hỏng. Có khuyến cáo rằng số các khoản mục dữ liệu có thể được truyền qua một giao diện nên giữ ở bẩy hay ít hơn.

Hình 4-2-3 Giao diện giữa các mô đun



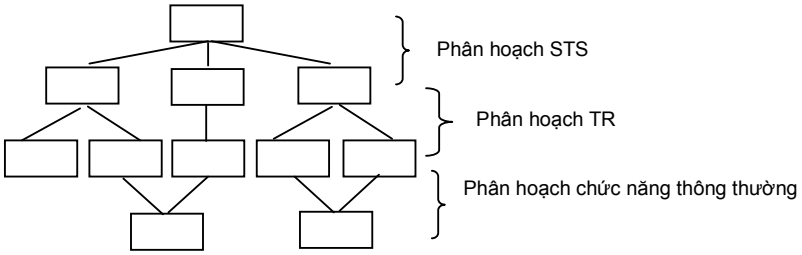
Số đường kết nối	Đưa vào từ các module mức cao	Đưa ra từ các module mức cao
1	Mã hàng, Số lượng sản phẩm theo yêu cầu, ngày vận chuyển theo yêu cầu	Trạng thái, thông điệp A, thông điệp B
2	Mã hàng	Số hàng tồn kho không dữ trữ
3	Mã hàng	Ngày các sản phẩm được lưu kho theo kế
hoạch,		số sản phẩm được lưu kho theo kế hoạch

(6) Xem xét việc phân hoạch lại mô đun

Nếu chương trình được phân hoạch thành các mô đun, dựa trên tiêu chí cho việc phân hoạch mô đun được cho với ước lượng không mong muốn, thì một trong những phương pháp phân hoạch được nêu trong (3) phải được chọn để làm lại nó theo cách trên xuống. Nếu làm như

vậy, nhiều phương pháp có thể được dùng tổ hợp với nhau khi cần.

Hình 4-2-4 Dùng một phương pháp phân hoạch tổ hợp với các phương pháp khác



<Cách thức từng phương pháp phân hoạch phải được dùng >

1. Phương pháp phân hoạch STS được dùng cho các mô đun mức cao.
2. Phương pháp phân hoạch TR được dùng cho các mô đun mức trung.
3. Phương pháp phân hoạch chức năng thường được dùng để tích hợp các mô đun chi tiết, mức thấp.

4.2.2 Các kĩ thuật phân hoạch mô đun điển hình

(1) Các kĩ thuật phân hoạch được thiết kế với chú ý tập trung vào luồng dữ liệu

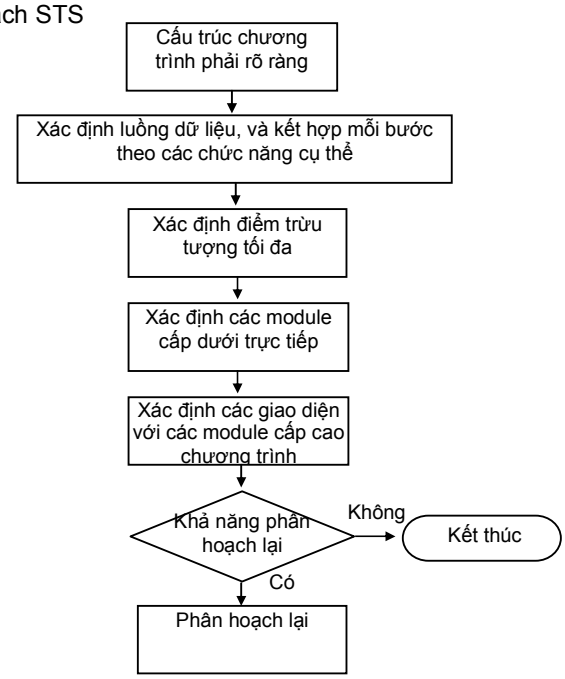
① Phương pháp phân hoạch STA

Về nguyên tắc, dữ liệu được xử lí qua ba bước: đưa vào, xử lí và đưa ra. Phương pháp phân hoạch STS (Source –Transform-Sink Nguồn-Biến đổi-Bể chứa) được thiết kế với chú ý tập trung vào luồng dữ liệu này, chương trình được chia thành ba phần như sau:

- Nguồn (chức năng xử lí đầu vào)
- Biến đổi (chức năng xử lí dữ liệu)
- Bể chứa (chức năng xử lí đầu ra)

Phương pháp này là thích hợp để giải quyết cho các mô đun mức cao có tất cả các chức năng đưa vào, xử lí và đưa ra.

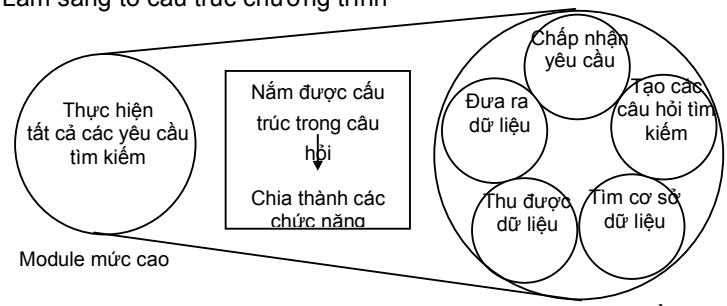
Hình 4-2-5 Thủ tục phân hoạch STS



<Thủ tục phân hoạch STS>

1. Trước hết cấu trúc chương trình được làm sáng tỏ.
   Cấu trúc chương trình được làm sáng tỏ với chú ý chính dành cho các chức năng, như được đưa ra trong Hình 4-2-6. Các chức năng nên được gộp nhóm lại sao cho số các chức năng là từ ba tới mười.

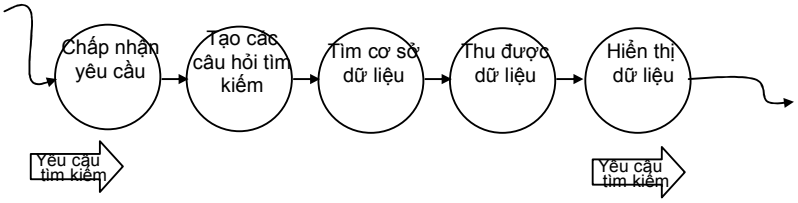
Hình 4-2-6 Làm sáng tỏ cấu trúc chương trình



2. Luồng dữ liệu vào và ra được nhận diện và từng bước trong luồng này được liên kết với các chức năng xác định.

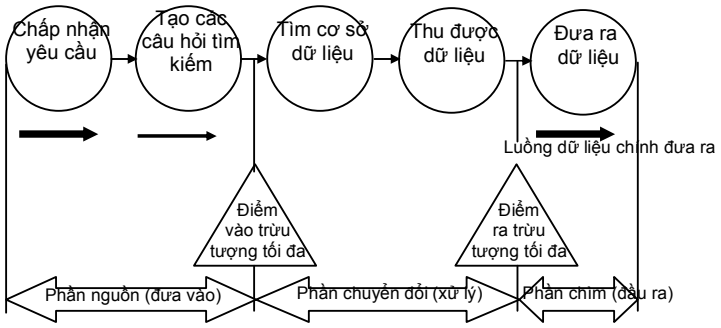
Từng bước trong luồng dữ liệu này trong chương trình được liên kết và móc nối lại, dùng các mũi tên (sơ đồ bọt) để tạo ra luồng dữ liệu chính cho vào và ra, như được vẽ trong Hình 4-2-7.

Hình 4-2-7 Liên kết từng bước trong luồng dữ liệu với các chức năng xác định



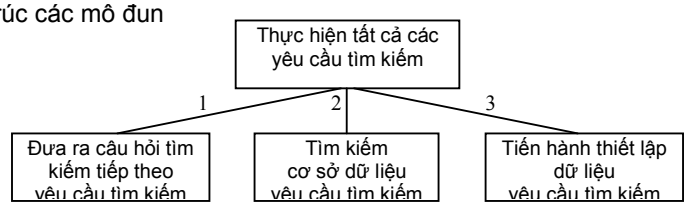
3. Nhận diện điểm trừu tượng tối đa
   Nhận diện điểm trừu tượng nơi dữ liệu có thể không còn được xem xét là dữ liệu vào (điểm vào trừu tượng tối đa), và điểm mà dữ liệu bắt đầu thành hình như dữ liệu đưa ra (điểm đưa ra trừu tượng tối đa), như được nêu trong Hình 4-2-8.

Hình 4-2-8 Điểm trừu tượng tối đa



3. Xác định các mô đun cấp dưới trực tiếp
   Cấu trúc, và liên kết các mô đun (kể cả mô đun cấp cao) đã được phân hoạch thành cái vào, xử lí và cái ra, như được nêu trong Hình 4-2-9.

Hình 4-2-9 Cấu trúc các mô đun



5. Xác định các giao diện với các mô đun cấp cao
   Xác định giao diện với các mô đun cấp cao (giao diện giữa các mô đun). (Xem Hình 4-2-10.)

Hình 4-2-10 Xác định các giao diện giữa các mô đun

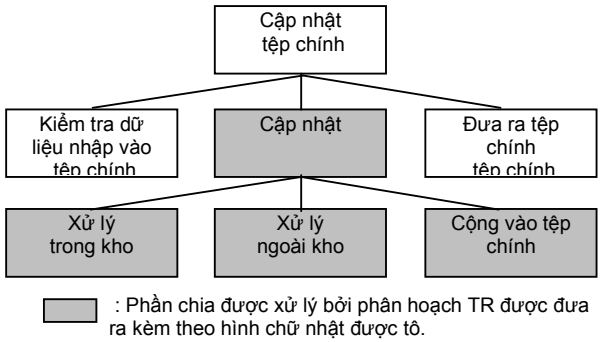
	Đầu vào	Đầu ra
1 2	(không)	- Tìm kiếm câu hỏi - Bản yêu cầu - Địa chỉ cuối - Mã lỗi - Bản thông tin - Mã lỗi
3	. Bản thông tin . Bản yêu cầu . Địa chỉ cuối	- Mã lỗi

6. Kiểm tra khả năng phân hoạch lại
   Kiểm tra xem liệu có mô đun nào phải được phân hoạch lại không. Nếu có, tiếp tục phân hoạch các mô đun.

② Phương pháp phân hoạch TR (phương pháp phân hoạch giao tác)

Nếu kiểu giao tác có thể được xác định bằng kiểu của dữ liệu, thì nên dùng phương pháp phân hoạch TR gộp nhóm (làm mô đun hoá) cho các mô đun theo kiểu giao tác. Phương pháp này phụ thuộc dữ liệu.

Hình 4-2-11 Các mô đun được phân hoạch dùng phương pháp phân hoạch TR

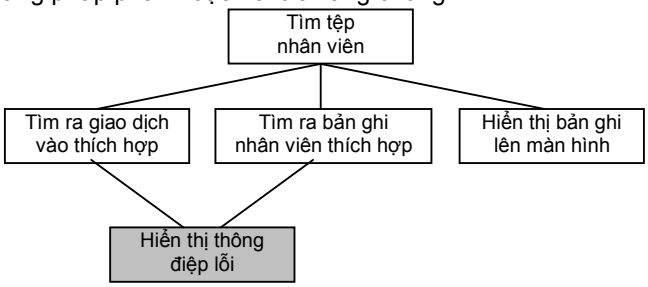


Trong ví dụ được nêu trong Hình 4-2-11, các kiểu giao tác được phân hoạch theo các kiểu dữ liệu vào (xử lí trong kho, xử lí ngoài kho, bổ sung vào tệp chính). Khi cần xử lí một kiểu dữ liệu nào đó, một giao tác tương ứng của những giao tác đã được phân hoạch này sẽ được lựa ra.

③ Phương pháp phân hoạch chức năng chung

Nếu có các mô đun có các chức năng chung, thì phương pháp này được dùng. (Xem Hình 4-2-12.)

Hình 4-2-12 Phương pháp phân hoạch chức năng chung



(2) Kĩ thuật phân hoạch hướng cấu trúc dữ liệu

Các mô đun được thiết kế bằng việc thiết lập quan hệ cấu trúc của mô đun với cấu trúc của dữ liệu vào và ra.

① Phương pháp Jackson

Với phương pháp Jackson, các mô đun được phân hoạch bằng việc lập mối quan hệ giữa cấu trúc của chương trình với cấu trúc của dữ liệu vào và ra.

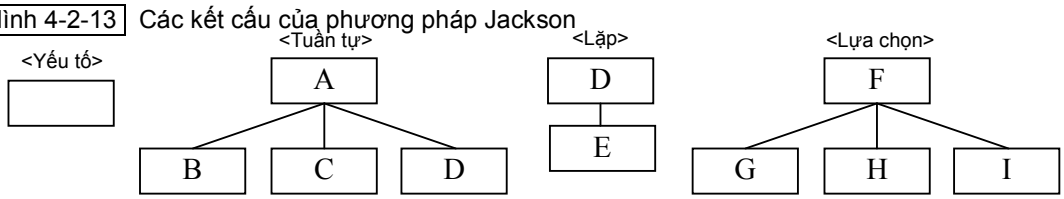
<Nhiệm vụ phân hoạch của phương pháp Jackson>

1. Xác định cấu trúc của dữ liệu vào và ra.
2. Tìm mối quan hệ một-một giữa các kết cấu trong cấu trúc của dữ liệu vào và ra. Nếu không thể tìm được, thì một cấu trúc dữ liệu trung gian phải được thiết lập.
3. Tạo ra cấu trúc chương trình dựa trên cấu trúc của dữ liệu ra.
4. Kiểm chứng cấu trúc chương trình bằng việc tham chiếu tới cấu trúc dữ liệu vào.

<Các kết cấu của phương pháp Jackson>

Trong trường hợp của phương pháp Jackson, một cấu trúc dữ liệu cũng như cấu trúc chương trình được xây dựng nên bằng việc dùng ba kết cấu, "tuần tự," "lặp" và "tuyển lựa" được định nghĩa tương ứng theo các phần tử cơ sở.

Hình 4-2-13 Các kết cấu của phương pháp Jackson

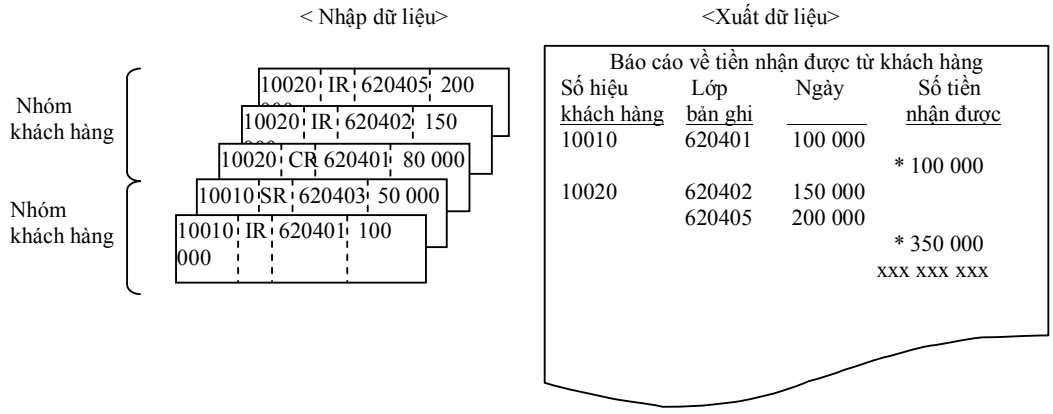


- Yếu tố: Các kết cấu không thể được phân chia thêm nữa
  Ví dụ: Khoản mục dữ liệu, câu lệnh, v.v...
- Tuần tự: Kết cấu bao gồm các kết cấu con; mỗi kết cấu con xuất hiện tuần tự chỉ một lần.
  Ví dụ: Bản ghi (nhiều khoản mục dữ liệu), trình xử lí tuần tự, v.v.
- Lặp (*): Một kết cấu xuất hiện lặp đi lặp lại.
  Ví dụ: Tệp tuần tự, câu lệnh PERFORM, v.v...
- Tuyển lựa (°): Bao gồm nhiều kết cấu con; một trong chúng được lựa lấy.
  Ví dụ: Nợ-Có, câu lệnh EVALUATE v.v...

<Cách phân hoạch mô đun>

Dùng phương pháp Jackson, các mô đun được phân hoạch như được nêu trong Hình 4-2-14.

Hình 4-2-14 Dữ liệu vào và ra



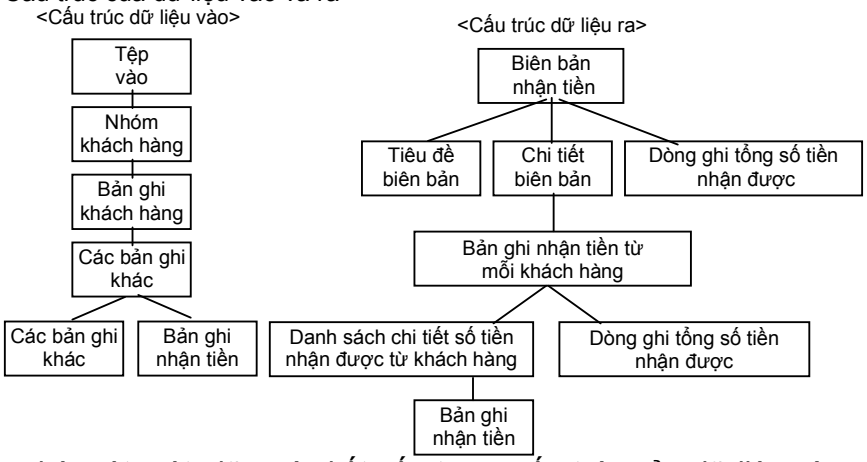
- Xác định cấu trúc của dữ liệu vào và ra.

  Dữ liệu vào bao gồm những điều sau, như được nêu trong Hình 4-2-14:
  - Tệp vào: Cùng nhóm khách hàng được lặp lại.
  - Nhóm khách hàng: Cùng bản ghi nhóm khách hàng được lặp lại.
  - Kiểu của bản ghi khách hàng: Bản ghi tiền nhận được và các bản ghi khác

  Dữ liệu ra bao gồm những điều sau:
  - Dữ liệu ra: Bản ghi về tiền nhận được từ mỗi khách hàng được lặp lại.
  - Bản ghi về tiền nhận được: Danh sách chi tiết tiền nhận được từ từng khách hàng, và tổng số tiền nhận được từ khách hàng được lặp lại.
  - Danh sách chi tiết về tiền nhận được từ khách hàng: Bản ghi về tiền nhận được được lặp lại.

  Do đó, các cấu trúc của dữ liệu vào và ra có thể được xác định như được nêu trong Hình 4-2-15.

Hình 4-2-15 Cấu trúc của dữ liệu vào và ra

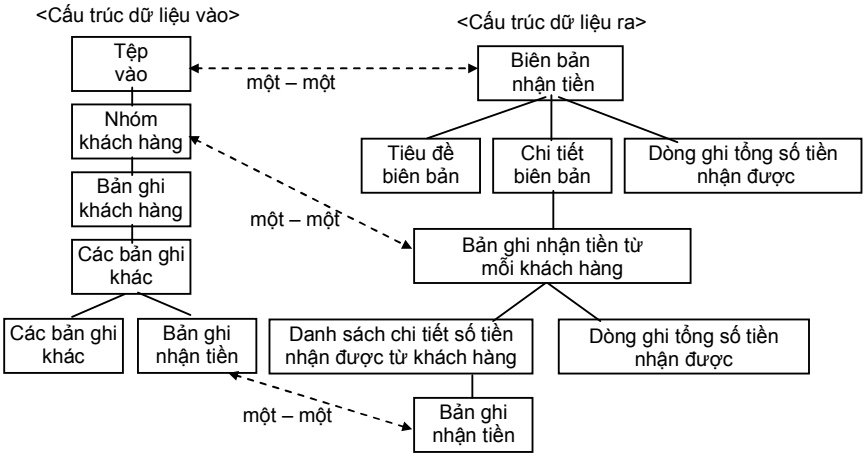


- Tìm mối quan hệ một-một giữa các kết cấu trong cấu trúc của dữ liệu vào và ra (xem Hình 4-2-16). Nếu không có quan hệ như vậy, phải thiết lập ra một cấu trúc dữ liệu trung gian.

  Trong trường hợp của Hình 4-2-15, có ba quan hệ một-một:
  - Tệp vào và báo cáo về tiền nhận được

- Nhóm khách hàng và bản ghi về tiền nhận được từ từng khách hàng
- Một bản ghi tiền nhận được và bản ghi khác về tiền nhận được

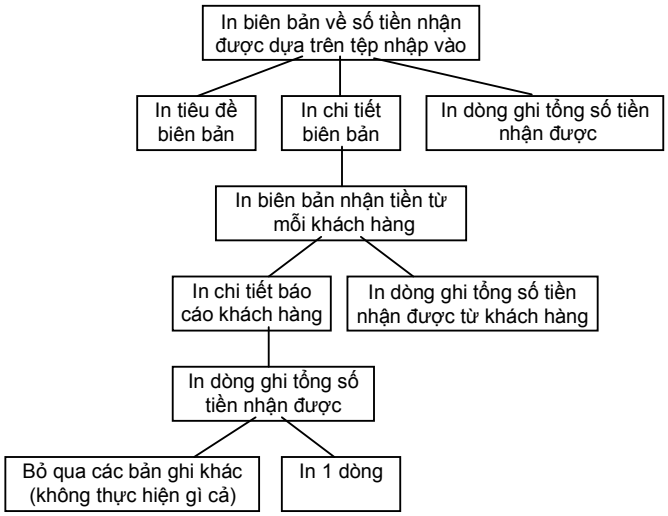
Hình 4-2-16 Mối quan hệ một-một giữa các phần tử tích hợp



- Tạo ra cấu trúc chương trình dựa trên cấu trúc của dữ liệu ra.

  Tạo ra cấu trúc chương trình dựa trên mối quan hệ một-một giữa các kết cấu. (Xem Hình 4-2-17.) Về nguyên tắc, cấu trúc của chương trình phải có quan hệ với dữ liệu đưa ra, và dữ liệu đưa vào được dùng cho cả việc kiểm chứng và sửa cấu trúc chương trình. (Chỉ dẫn "Bỏ qua các bản ghi khác - Ignoring other records" là kết quả thu được sau khi một phần của cấu trúc chương trình đã được sửa lại.)

Hình 4-2-17 Cấu trúc chương trình



② Phương pháp Warnier

Phương pháp Warnier là kĩ thuật thiết kế mô đun có cấu trúc dựa trên lí thuyết tập hợp. Nó được sử dụng rộng rãi để phân hoạch các mô đun cho việc xử lí tệp và các ứng dụng nghiệp vụ khác.

<Đặc trưng của phương pháp Warnier>

- Phân tích dữ liệu là cơ sở của phương pháp này.
- Phương pháp này dựa trên "khi nào, ở đâu và bao nhiêu lần." Việc phân tích được thực

hiện theo cách trên xuống.

- Trong khi phương pháp Jackson được thiết kế để làm việc cấu trúc chủ yếu dựa trên dữ liệu đưa ra, thì phương pháp Warnier được thiết kế để làm việc cấu trúc dựa trên dữ liệu đưa vào.

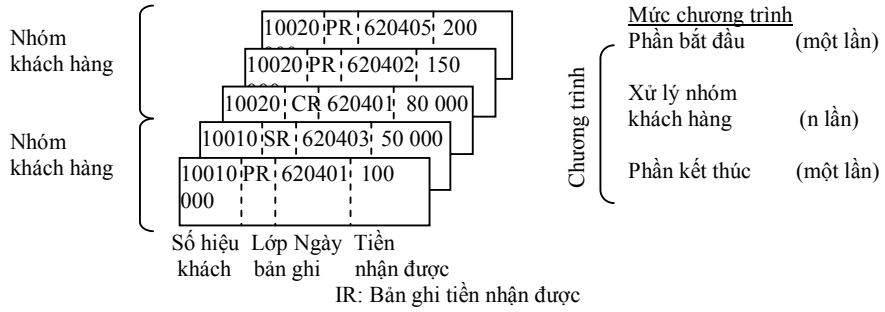
Các nhiệm vụ phân hoạch của phương pháp Warnier được nêu dưới đây. Xem như một ví dụ, lấy lại ví dụ được vẽ trong Hình 4-2-14.

<Các nhiệm vụ phân hoạch của phương pháp Warnier>

1. Tìm mối quan hệ một-một giữa cấu trúc của dữ liệu đưa vào và cấu trúc logic của chương trình.

Trước hết so sánh cấu trúc của dữ liệu vào với cấu trúc logic của chương trình.

Hình 4-2-18 So sánh cấu trúc của dữ liệu vào với cấu trúc logic của chương trình



Cùng cấu trúc như được nêu trong Hình 4-2-14 được tham khảo tới ở đây. Cấu trúc logic của chương trình tương ứng bao gồm các tập con sau:

- Phần bắt đầu
- Phần nhóm khách hàng
- Phần kết thúc

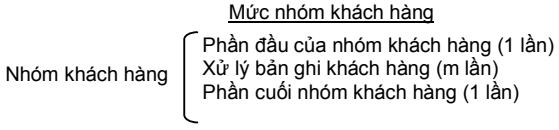
(N: số lần lặp, 0 và 1: hoặc được lựa)

Ví dụ được nêu trong Hình 4-2-18 chỉ ra rằng việc xử lí cho nhóm khách hàng tiếp tục cho tới khi việc xử lí cho tất cả các tệp được hoàn tất.

2. Chia ra các tập con theo cách từ trên xuống.

Nhóm khách hàng đã được chia thêm ra được nêu trong Hình 4-2-19.

Hình 4-2-19 Chia ra các tập con (bước 1)



Cấu trúc logic của nhóm khách hàng bao gồm các tập con sau:

- Phần bắt đầu của một nhóm khách hàng
- Phần bản ghi khách hàng
- Phần kết thúc của nhóm khách hàng

Phần bản ghi khách hàng đã được phân chia thêm được nêu trong Hình 4-2-20.

Hình 4-2-20 Chia ra các tập con (bước 2)

Mức bản ghi khách hàng

Bản ghi khách hàng
- Phần bắt đầu của bản ghi khách hàng (1 lần)
- Liệu nó có là bản ghi về tiền nhận được hay không(không hoặc 1 lần)
- Phần kết thúc nhóm khách hàng (1 lần)

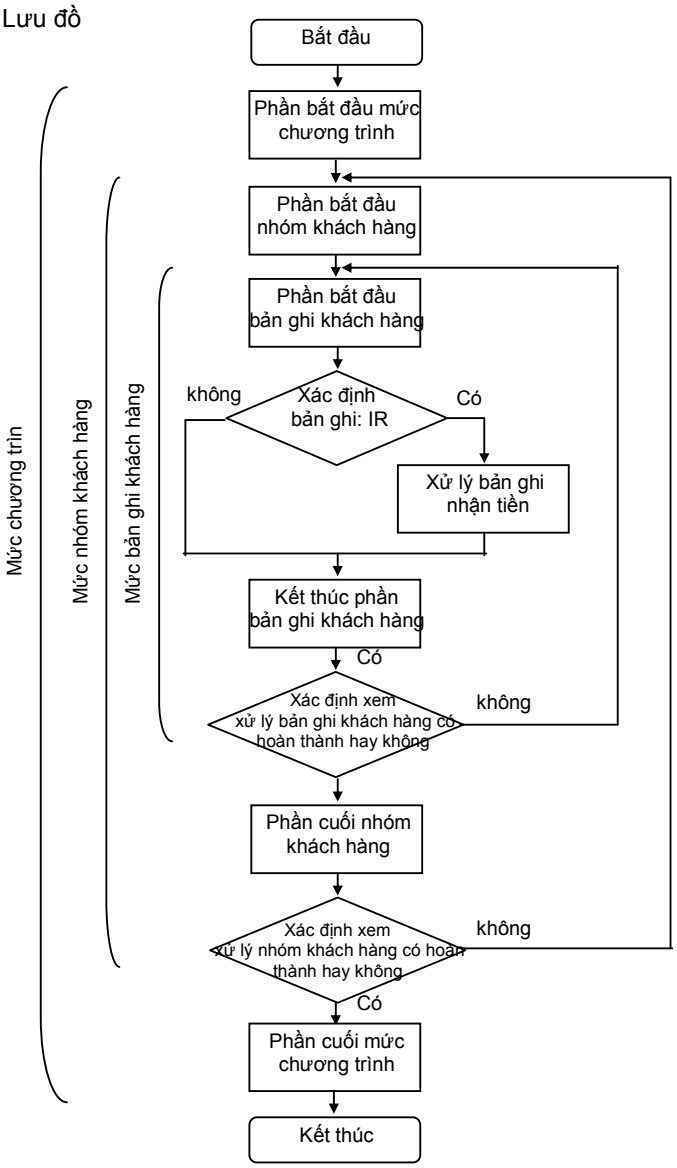
Cấu trúc logic của phần bản ghi khách hàng bao gồm các tập con sau:
- Phần bắt đầu của bản ghi khách hàng
- Liệu nó có là bản ghi về tiền nhận được hay không
- Phần kết thúc của bản ghi khách hàng

Vì tập con "liệu nó có là bản ghi về tiền nhận được hay không " không thể được phân chia thêm nữa, nên việc phân hoạch dừng lại tại mức này.

3. Vẽ lưu đồ.

Lưu đồ được vẽ ra dựa trên Hình 4-2-18, 4-2-19 và 4-2-20 được nêu trong Hình 4-2-21.

Hình 4-2-21 Lưu đồ



(3) Dùng tổ hợp các kĩ thuật phân hoạch

Kĩ thuật phân hoạch cơ sở được dùng cho thiết kế cấu trúc là phương pháp STS. Trong bước đầu tiên của việc phân hoạch chương trình, phương pháp này bao giờ cũng được dùng, bởi vì việc phân hoạch STS cho phép người lập trình nhận diện cấu trúc duy nhất của chương trình để làm việc tiếp. Nó cũng làm cho người lập trình có khả năng hiểu cách thức dữ liệu chảy qua tại từng bước của việc xử lí dữ liệu, và cách nó được biến đổi trong khi được xử lí bởi từng chức năng. Trong phương pháp phân hoạch STS, các chức năng được chia thành các chức năng cấp dưới.

Nếu phương pháp phân hoạch STS không thể áp dụng được, thì phải dùng phương pháp khác. Phân hoạch STS không làm việc tốt với các mô đun phụ thuộc dữ liệu – các mô đun mà khó rút ra cấu trúc chương trình trực tiếp. Trong trường hợp này phải dùng phân hoạch TR. Trong phương pháp phân hoạch TR, các chức năng được chia thành các chức năng cấp dưới tương đương.

Khi các mô đun được phân hoạch ra, số các mô đun liên tục tăng lên. Tuy nhiên, nó không tăng vô hạn; khi các mô đun được phân hoạch tới một giới hạn nào đó thì các mô đun ở đầu xa nhất chỉ là các câu lệnh. Mặc dầu đây là trường hợp cực đoan, việc phân hoạch nên dừng lại tại điểm nào đó. Tiêu chuẩn được nêu dưới đây được dùng để xác định khi nào thì dừng việc phân hoạch:

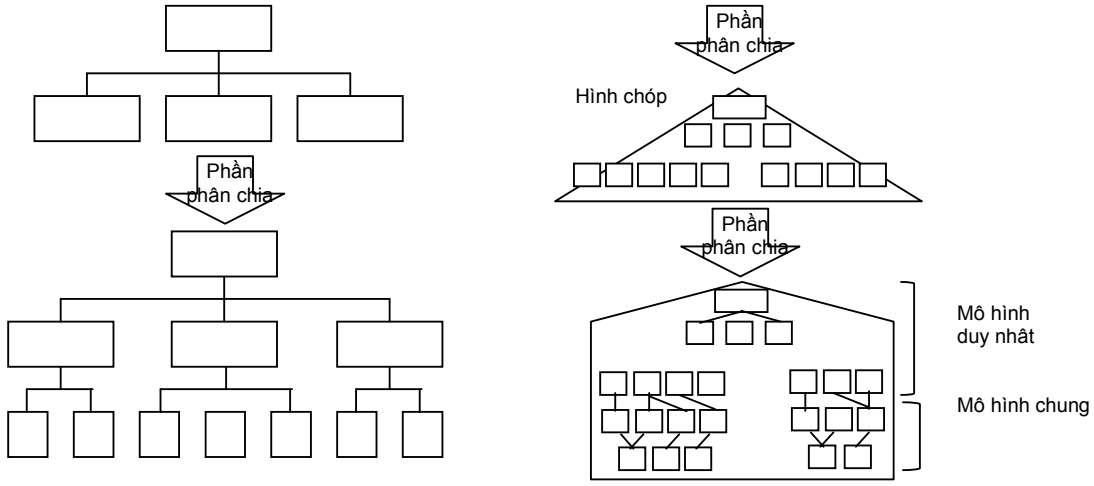
① Trước khi bạn bắt đầu phân hoạch một mô đun, hãy nhìn trước vào cấu trúc logic của mô đun. Nếu bạn có thể nhìn trước nó một cách dễ dàng, thì mô đun được ước lượng chứa quãng năm mươi câu lệnh. Do đó, bạn có thể xác định rằng bất kì việc phân hoạch nào thêm cũng đều không cần thiết.

② Nếu bạn không còn có thể phân hoạch được mô đun thành các mô đun cấp dưới trực tiếp có độ bền về mặt chức năng (sẽ giải thích dưới đây), thì bạn có thể xác định rằng bất kì việc phân hoạch nào thêm nữa cũng là không cần thiết.

③ Nếu bạn phân vân giữa việc dừng lại và việc tiếp tục phân hoạch, thì bạn nên tiếp tục phân hoạch. Nếu việc phân hoạch của bạn quá nhiều, bạn có thể tổ chức lại các mô đun, tuy nhiên sửa chữa chương trình được phân hoạch quá nhiều còn dễ hơn là chương trình được phân hoạch không đủ.

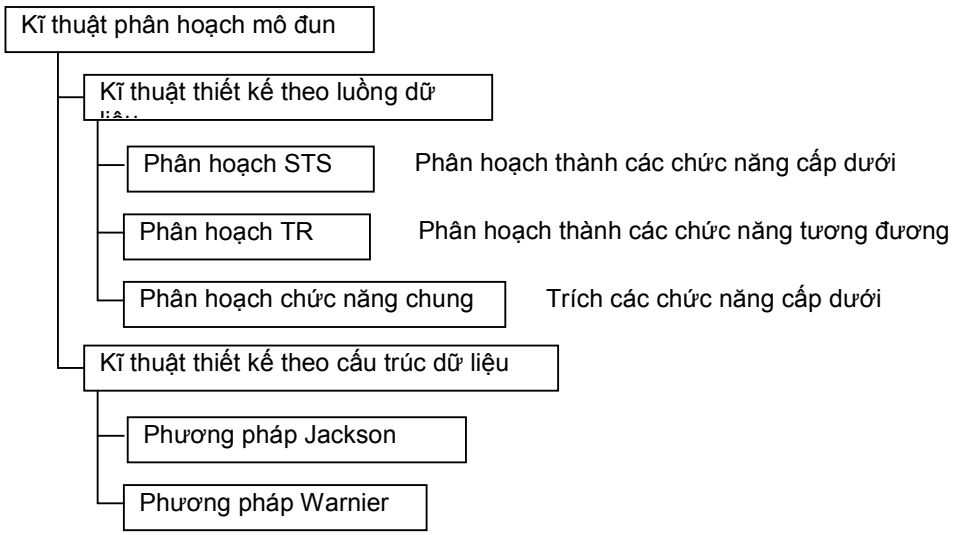
Cấu trúc mô đun thay đổi từ hình chóp nhọn sang hình dáng mái đền thờ Hồi giáo, như được nêu trong Hình 4-2-22. Điều này là do việc phân hoạch chức năng thông thường được thực hiện ở bước cuối cùng của việc phân hoạch mô đun.

Hình 4-2-22 Hình dáng của cấu trúc mô đun



Hình 4-2-23 cung cấp tổng quan về các kĩ thuật phân hoạch mô đun, hay các kĩ thuật thiết kế chương trình đã được mô tả cho tới nay.

Hình 4-2-23 Các kĩ thuật phân hoạch mô đun



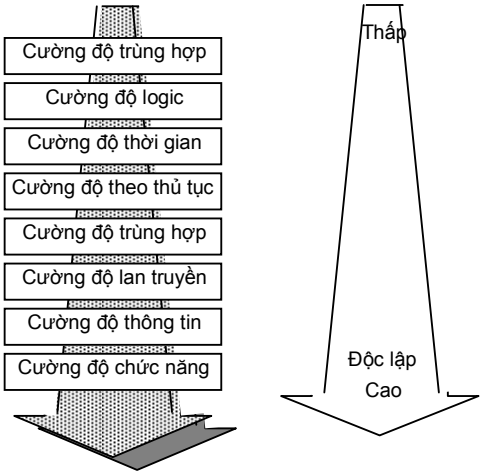
4.2.3 Tiêu chí cho việc phân hoạch mô đun

Công việc thiết kế chương trình phức tạp có thể được làm đơn giản hoá bằng việc phân hoạch chương trình thành các mô đun, như đã giải thích trong mục trước. Trong thiết kế và sửa đổi, từng phần của chương trình phải đảm bảo tính độc lập của mô đun, để cho tất cả các mô đun có thể thực hiện các chức năng đã nêu mà không ảnh hưởng lẫn nhau. Hai điểm này, việc làm đơn giản hoá thiết kế chương trình và tính độc lập của mô đun, là những mục tiêu chủ yếu phải được đạt tới trong thiết kế có cấu trúc.
Có hai cách đo được dùng để đánh giá tính độc lập của mô đun: độ bền mô đun chỉ ra sức bền của mối quan hệ giữa các mô đun, và việc gắn nối mô đun chỉ ra mối quan hệ giữa các mô đun

(1) Độ bền mô đun

Độ bền mô đun là một trong những hướng dẫn mà ta có thể tham chiếu tới khi xác định các mô đun trong tiến trình thiết kế. Các kiểu độ bền mô đun được xem xét từ khía cạnh của việc dùng lại mô đun, khuynh hướng lỗi, tính độc lập, tính bảo trì được, tính mở rộng được v.v.. Đặc biệt, chúng được xem xét theo trật tự những hiệu quả không mong muốn cho tới hiệu quả được mong muốn, được tạo ra bởi từng kiểu độ bền mô đun.

Hình 4-2-24 Độ bền mô đun



① Độ bền trùng hợp ngẫu nhiên

Nếu bất kì điều kiện nào trong những điều kiện sau mà áp dụng được cho một mô đun bạn đang kiểm tra, độ bền của mô đun như vậy được gọi là độ bền trùng hợp ngẫu nhiên :

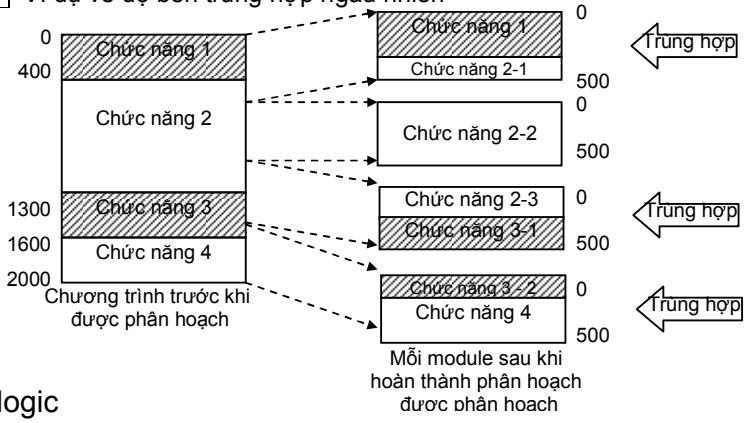
- Các chức năng của mô đun không thể được xác định.
- Mô đun có nhiều chức năng không liên quan lẫn nhau.

Mặc dầu người lập trình sẽ không chủ ý tạo ra mô đun như vậy, điều đó có thể được tạo ra một cách không chủ ý trong những hoàn cảnh sau đây:

- Nếu mô đun được phân hoạch thành những phần tuỳ tiện để gọi chèn lấp.
- Nếu người lập trình phải đáp ứng yêu cầu rất nghiêm khắc, chẳng hạn "số các câu lệnh trong từng mô đun phải là 50 tới 60".

Mô đun được tạo ra trong những hoàn cảnh như vậy không thể được dùng lại chút nào; nó có thể gây ra việc giảm sút tính bảo hành được cũng như tính mở rộng được của chương trình. Người ta cho rằng chương trình bị phân hoạch thành các mô đun có độ bền trùng hợp ngẫu nhiên còn tồi hơn chương trình không được phân hoạch chút nào.
Hình 4-2-25 đưa ra một ví dụ về độ bền trùng hợp ngẫu nhiên .

Hình 4-2-25 Ví dụ về độ bền trùng hợp ngẫu nhiên



② Độ bền logic

Mô đun có hai hay nhiều chức năng có liên quan, và việc thực hiện một trong những chức năng này do mô đun gọi lựa ra, về thuật ngữ được gọi là độ bền logic. Kiểu mô đun này

thực hiện các chức năng qua một giao diện. Hình 4-2-26 đưa ra kiểu mô đun này.

Hình 4-2-26 Mô đun với độ bền logic (ví dụ)

Tên module:	
Các đối được chấp nhận: 4 đối	
Đối 1: Mã chức năng (1 số)	Nếu đối là 0: Xóa bảng Nếu đối là 1: Thêm các mục vào bảng Nếu đối là 2: Xóa các mục từ bảng Nếu đối là 3: Tìm bảng Nếu đối là 4: Sao chép bảng tới tệp kiểm tra
Đối 2: Tên (4 số)	Các chức năng của đối là đối nhập nếu các chức năng 1,2 và 3 được sử dụng Các chức năng của đối là đối giả nếu các chức năng 0 và 4 được sử dụng
Đối 3: (8 số)	Các chức năng của đối là đối nhập nếu chức năng 1 được sử dụng Các chức năng của đối là đối xuất nếu chức năng 3 được sử dụng Các đối khác là đối giả
Đối 4: Cờ lỗi (1 số)	Đối ra (trừ chức năng 0) 0: Không lỗi (kết thúc bình thường) Chức năng 1: 1 - bảng đầy đủ, 2 – các mục chồng chéo lên nhau Chức năng 2 và 3: 1 – không có mục nào Chức năng 4: 1 - lỗi vào / ra, 2 - kết thúc tệp, 3 – không có dữ liệu

Với mô đun có độ bền logic việc giải quyết nó có chút ít rắc rối hơn bởi những lí do được nêu sau đây:

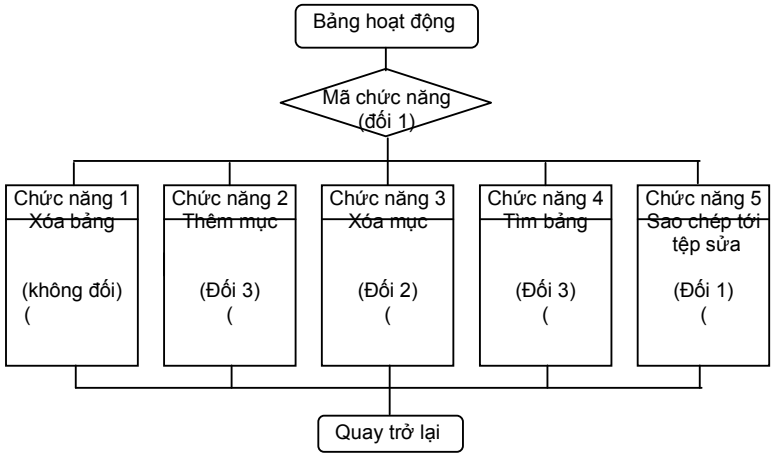
- Đối được diễn giải theo cách khác, tuỳ theo chức năng được gọi.
- Mặc dầu một đối nào đó bị bỏ qua bởi chức năng nào đó, nó không thể bị bỏ đi được.
- Cho dù chỉ một chức năng được dùng, cũng cần thừa nhận các chức năng khác.

Cũng vậy, nếu một chức năng được một mô đun dùng mà phải bị sửa đổi, thì tất cả các mô đun khác dùng mô đun đặc biệt này cũng đều phải được sửa đổi mặc dầu chúng là không liên quan tới chức năng được thay đổi.

Quan niệm tất cả các nhiệm vụ xử lí được thực hiện theo bảng trên có thể được tổ hợp vào trong một mô đun là đúng và hợp lí. Để đưa quan niệm này vào thực hành, điều đáng mong muốn là dùng một mô đun có độ bền thông tin .

Hình 4-2-27 đưa ra một ví dụ về độ bền logic .

Hình 4-2-27 Độ bền logic (ví dụ)

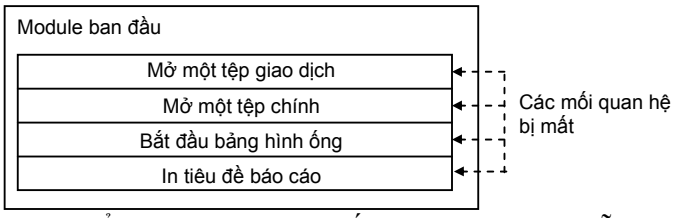


③ Độ bền thời gian

Mô đun thực hiện nhiều chức năng tuần tự được gọi là mô đun có độ bền thời gian. Tuy nhiên, các chức năng này có mối quan hệ yếu với nhau. Mô đun khởi đầu hay mô đun kết thúc thuộc kiểu mô đun này.

Hình 4-2-28 đưa ra mô đun có độ bền thời gian .

Hình 4-2-28 Độ bền thời gian (ví dụ)

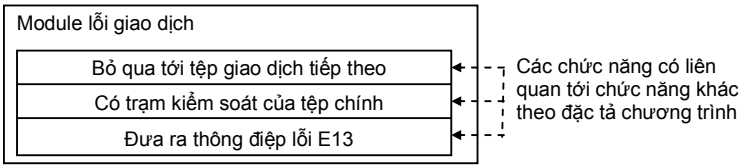


Các chức năng trong kiểu mô đun này có mối quan hệ lỏng lẻo lẫn nhau. Tuy nhiên, có một mô đun thực hiện hai hay nhiều chức năng có mối quan hệ mạnh. Nếu một trong các chức năng này là để khởi đầu một bảng, như được nêu trong Hình 4-2-28, một trình thao tác cho bảng này nên tồn tại điểm ở xa với mô đun khởi đầu trong chương trình. Trong mối quan hệ của nó với các mô đun khác (các mô đun không tường minh), khó mà phân biệt được mô đun có độ bền thời gian với các mô đun khác.
Do đó, một mô đun có độ bền thời gian hầu như không đóng góp gì mấy cho tính độc lập của chương trình.

④ Độ bền thủ tục

Mô đun có độ bền thủ tục là tương tự như mô đun có độ bền thời gian vì nó thực hiện nhiều chức năng tuần tự. Tất cả các quan hệ tuần tự giữa các chức năng, tuy vậy, lại được tổ chức bên trong thủ tục giải quyết vấn đề. Kiểu mô đun này được gọi là mô đun có độ bền thủ tục. Hình 4-2-29 nêu ra một ví dụ về mô đun có độ bền thủ tục.

Hình 4-2-29 Độ bền thủ tục

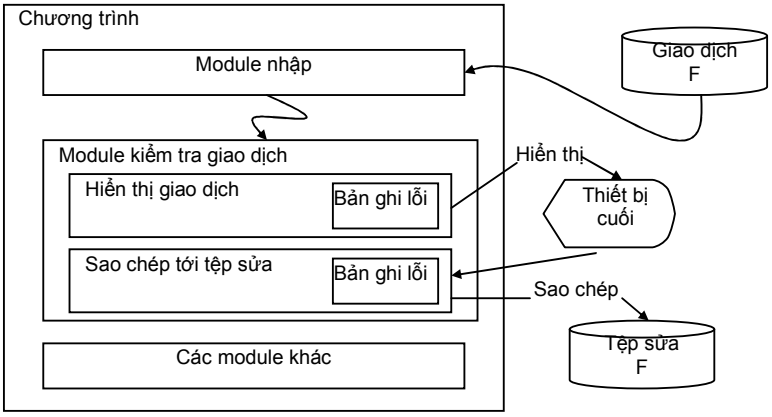


Mối quan hệ giữa các chức năng của một mô đun có độ bền thủ tục có hơi mạnh hơn mối quan hệ giữa các chức năng của mô đun có độ bền thời gian. Nếu chúng không được xác định, thì chúng khó mà có thể thấy rõ được. Bởi vì mức của độ bền thủ tục vào quãng giữa trong cấp bậc 7 mức, nên chẳng có gì đặc biệt để mà chúng ta phải coi thường, hoặc là xứng đáng nhắc tới một cách đặc biệt về kiểu mô đun này.

⑤ Độ bền trao đổi

Trong một mô đun có độ bền trao đổi, mối quan hệ tuần tự giữa tất cả các chức năng được diễn giải như một thủ tục giải quyết vấn đề. Điều này là giống như mô đun có độ bền thủ tục. Một mô đun được thiết kế ra với đặc trưng này, cộng thêm với đặc trưng là có mối quan hệ dữ liệu giữa tất cả các chức năng, được gọi là mô đun có độ bền trao đổi. Tức là, sự khác biệt nổi bật giữa mô đun có độ bền trao đổi và mô đun có độ bền thủ tục là ở chỗ tất cả các chức năng đều tham chiếu tới cùng dữ liệu.
Nếu một đặc tả mô đun nêu ra rằng một giao tác không thích hợp được hiển thị trên thiết bị cuối và được sao chép lên tệp kiểm định, chẳng hạn, như được nêu trong Hình 4-2-30, thì mô đun chứa chức năng hiển thị trên thiết bị cuối và mô đun chứa chức năng sao chép lên tệp kiểm định là các mô đun có độ bền trao đổi .

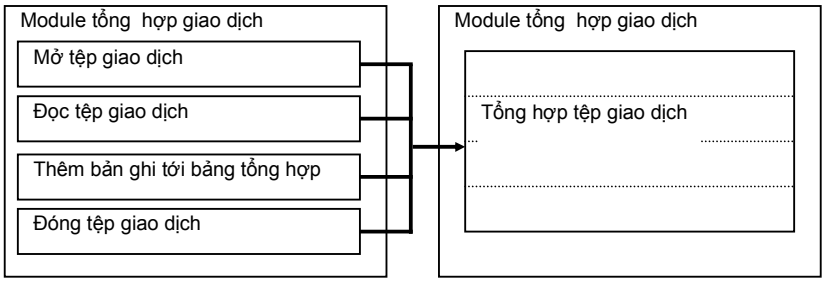
Hình 4-2-30 Độ bền trao đổi (ví dụ)



⑥ Độ bền chức năng

Đáng ra độ bền thông tin nên được nói tới ở đây, nếu chúng ta định tuân theo trật tự của độ bền, nhưng bây giờ chúng ta mô tả mô đun có độ bền chức năng trước. Độ bền chức năng là mạnh nhất trong tất cả các kiểu độ bền mô đun. Một mô đun thực hiện một chức năng duy nhất được gọi là mô đun có độ bền chức năng. Một mô đun không thể được phân loại thành các kiểu độ bền trùng hợp, logic, thời gian, thủ tục và trao đổi có thể được xem là mô đun có độ bền chức năng. Cho dù một mô đun có thực hiện hai hay nhiều chức năng, nó là mô đun có độ bền chức năng nếu chúng có thể được mô tả như một chức năng.
Hình 4-2-31 đưa ra một ví dụ về độ bền chức năng.

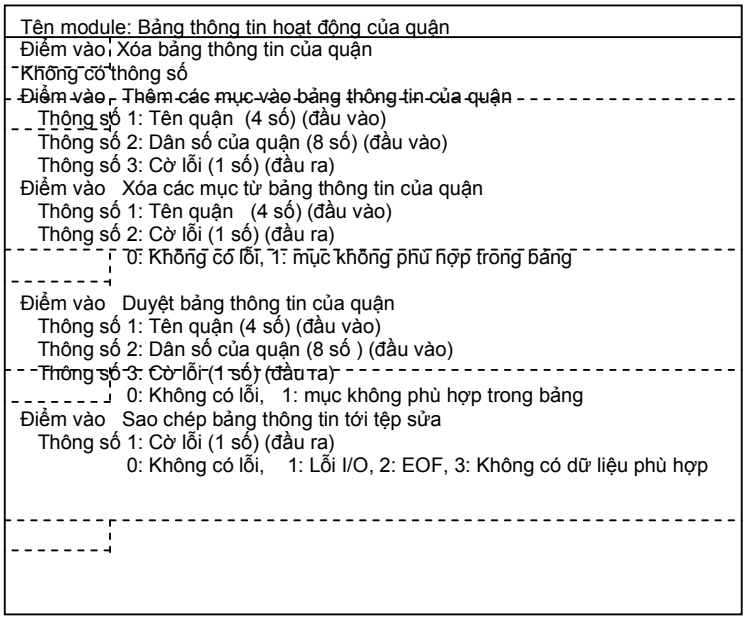
Hình 4-2-31 Độ bền chức năng (ví dụ)



⑦ Độ bền thông tin

Chương trình được thiết kế bằng việc dùng chỉ các mô đun có độ bền chức năng không nhất thiết có mức độc lập cao nhất. Chẳng hạn, ba mô đun có độ bền chức năng là mô đun thêm một khoản mục vào trong bảng kí hiệu, mô đun xoá khoản mục trong bảng kí hiệu và mô đun để duyệt bảng kí hiệu. (Xem Hình 4-2-32.)
Mặc dầu ba mô đun này là độc lập cao với các mô đun khác trong chương trình, chúng lại có quan hệ gần gũi lẫn nhau vì các chức năng của chúng đều phụ thuộc vào cấu trúc dữ liệu của bảng kí hiệu. Do đó, có thể dự đoán trước rằng nếu cần sửa đổi mô đun này, thì hai mô đun kia cũng phải được sửa đổi.

Hình 4-2-32 Mô đun có độ bền thông tin (ví dụ)



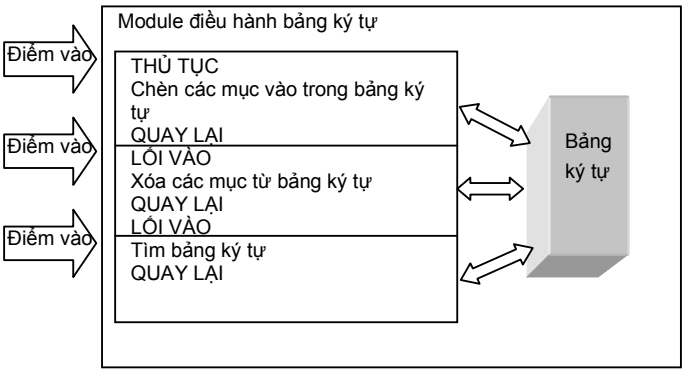
Trong ví dụ trên, tính độc lập của chương trình có thể được tăng lên bằng việc thay thế ba mô đun tương hỗ có độ bền chức năng, bằng một mô đun. Điều này được gọi là độ bền thông tin.

Đặc trưng của độ bền thông tin là:

- Có nhiều điểm vào.
- Mỗi điểm vào có một chức năng duy nhất.
- Tất cả các chức năng đều có liên quan tới một khái niệm, cấu trúc dữ liệu và tài nguyên được chứa bên trong mô đun .

Chủ định của độ bền thông tin là để chứa khái niệm, cấu trúc dữ liệu, tài nguyên v.v.. trong một mô đun và đạt tới việc che dấu thông tin.
Hình 4-2-33 nêu ra ví dụ về độ bền thông tin .

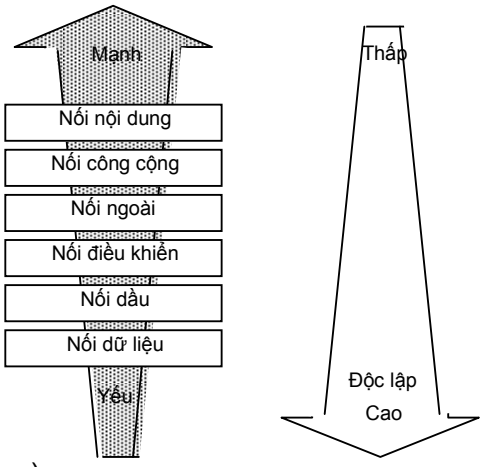
Hình 4-2-33 Độ bền thông tin (ví dụ)



(2) Nối mô đun

Độ bền mô đun quan tâm tới mối quan hệ bên trong mô đun. Mặt khác, nối mô đun quan tâm tới mối quan hệ giữa các mô đun. (Xem Hình 4-2-34.) Mối quan hệ giữa các mô đun ảnh hưởng tới tính độc lập của mô đun. Người ta đã xét thấy là mối quan hệ giữa các mô đun càng lỏng, thì mức độ độc lập của mô đun càng trở nên cao hơn.

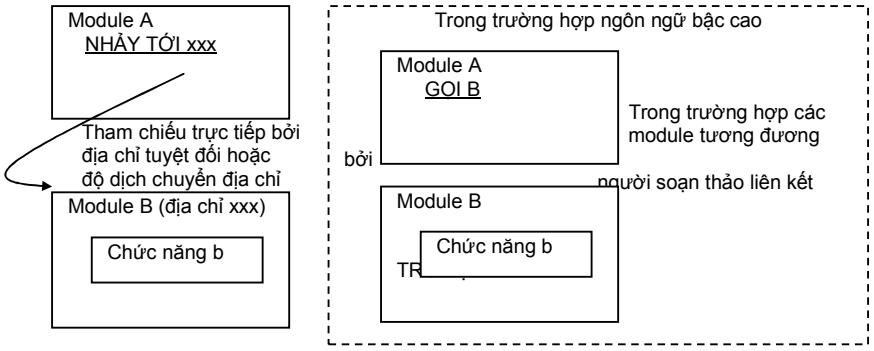
Hình 4-2-34 Nối mô đun



① Nối nội dung (content coupling)

Nếu một mô đun trực tiếp tham chiếu tới nội dung của mô đun khác, hay nhảy trực tiếp sang mô đun khác, thì mối quan hệ giữa hai mô đun này được gọi là nối nội dung. (Xem Hình 4-2-35.) Nếu mô đun này đã bị sửa đổi, thì cần sửa cả mô đun kia. Các mô đun có nối nội dung có thể được tạo ra bằng việc dùng hợp ngữ. Tuy nhiên, trong phần lớn các ngôn ngữ chúng không thể được tạo ra.

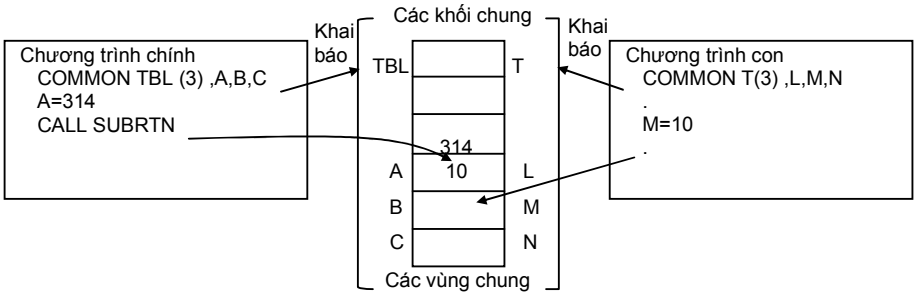
Hình 4-2-35 Nối nội dung (ví dụ)



② Nối công cộng (common coupling)

Mối quan hệ giữa các mô đun tham chiếu tới cấu trúc dữ liệu toàn cục (biến toàn cục) được gọi là nối công cộng. Chẳng hạn, có một khái niệm được gọi là khối common trong FORTRAN. Các biến được khai báo bằng câu lệnh COMMON để nối vùng bộ nhớ mà chương trình dùng. (Xem Hình 4-2-36.) Nối công cộng lấy theo tên của nó từ sự kiện là vùng bộ nhớ được làm thành chỗ chung để cho phép các mô đun dùng chung.

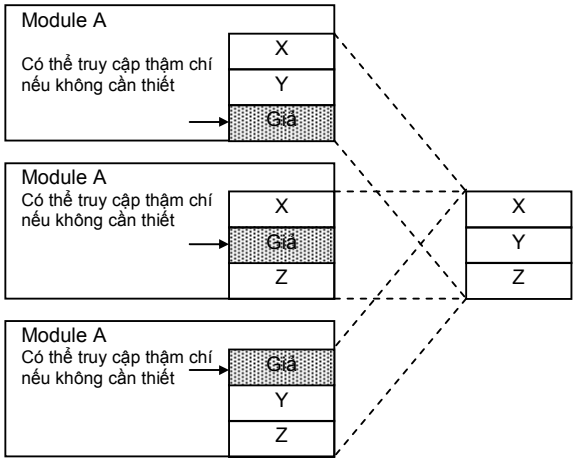
Hình 4-2-36 Nối công cộng (ví dụ 1)



Các vấn đề của nối công cộng được tóm tắt như sau:

- Dữ liệu toàn cục làm hư hại cho tính dễ đọc và dễ hiểu của chương trình.
- Cấu trúc dữ liệu toàn cục có thể gây cho các mô đun có vẻ liên quan lại trở thành phụ thuộc lẫn nhau.
- Các mô đun tham chiếu tới dữ liệu toàn cục thì khó dùng cho các chủ định khác. (Trong khi thực hiện xử lí song song hay xử lí đa nhiệm, tính nhất quán của vùng dữ liệu công cộng phải được tính tới.)
- Bởi vì các mô đun được nối qua các tên biến, nên khó dùng lại chúng cho chương trình mới.
- Nếu dữ liệu toàn cục là dữ liệu kiểu có cấu trúc, thì thậm chí lại còn khó sử dụng lại chúng hơn. (Tạo ra cấu trúc câm)
- Nếu cấu trúc dữ liệu toàn cục được dùng, thì nhiều dữ liệu hơn mức cần thiết sẽ bị phơi bầy ra cho các mô đun khác. (Xem Hình 4-2-37.)

Hình 4-2-37 Nối công cộng (ví dụ 2)



- Việc truy nhập dữ liệu bên trong chương trình không thể được quản lí đúng.

Nếu một phần trong một cấu trúc dữ liệu toàn cục bị thay đổi, thì mọi mô đun tham chiếu tới phần bị thay đổi đó của dữ liệu phải được dịch lại.

③ Nối ngoài (extern coupling)

Nếu có một nhóm các mô đun hoặc không nối nội dung hoặc không nối công cộng và tham chiếu tới cấu trúc dữ liệu toàn cục, thì mối quan hệ giữa các mô đun như vậy được gọi là nối ngoài . Mặc dầu việc nối ngoài quan tâm tới một tập các dữ liệu toàn cục khác nhau về định dạng và ý nghĩa, việc nối ngoài quan tâm tới tập dữ liệu có cùng một kiểu.

Dữ liệu cùng kiểu nghĩa là:

- Kiểu dữ liệu này không chứa kiểu dữ liệu khác; chẳng hạn dữ liệu số, dữ liệu kí tự, v.v...
- Bảng hay danh sách chỉ có một khoản mục
- Mảng có các phần tử với cùng nghĩa

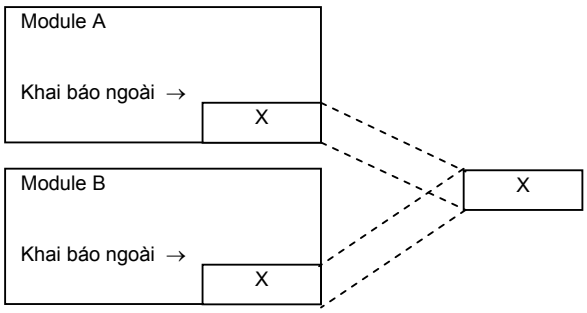
Trong nỗ lực của chúng ta để tìm ra giải pháp cho các vấn đề liên kết với việc nối công cộng, việc nối ngoài cho phép chúng ta cải thiện các điều kiện liên quan tới chỉ những vấn đề sau:

- Sự phụ thuộc không mong muốn giữa các mô đun
- Tạo ra cấu trúc câm
- Quá phơi bày dữ liệu

Cho dù chúng ta có thể thực hiện những cải tiến nào đó với các vấn đề trên, các điều kiện

vẫn còn không thoả mãn.

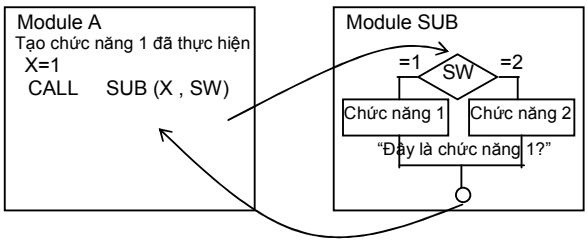
Hình 4-2-38 Nối ngoài



④ Nối điều khiển (control coupling)

Nếu có hai mô đun không nối nội dung cũng chẳng nối công cộng, chẳng nối ngoài và nếu mô đun này truyền các phần tử điều khiển cho mô đun kia, thì mối quan hệ giữa hai mô đun này được gọi là nối điều khiển. Khi mô đun này truyền một mã hàm hay khoá điều khiển cho mô đun kia có độ bền logic, thì mối quan hệ giữa chúng là nối điều khiển. Bởi vì một mô đun truyền dữ liệu điều khiển phải biết cấu trúc logic của mô đun kia, nên mức độ độc lập là không cao lắm. Mối quan hệ nối điều khiển bị hạn chế vào trường hợp mô đun này xem như nơi gửi dữ liệu, truyền các đối như phương tiện để điều khiển các chức năng và logic của mô đun kia xem như nơi nhận. Nếu mô đun này truyền đối mà không có mục đích nào được chỉ rõ cho mô đun kia thì mối quan hệ giữa hai mô đun này không thể được gọi là nối điều khiển, cho dù mô đun kia xem như nơi nhận dùng đối đó (dữ liệu điều khiển).

Hình 4-2-39 Nối điều khiển (ví dụ)

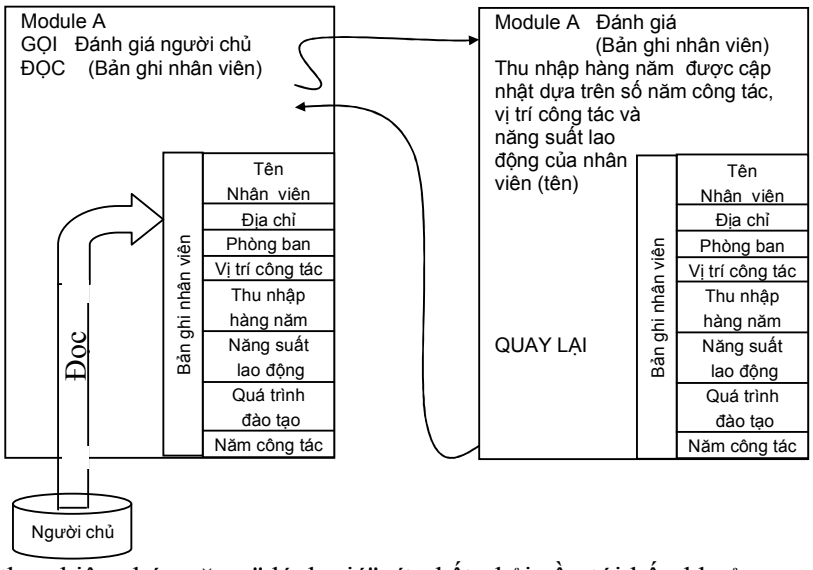


⑤ Nối dấu (stemp coupling)

Nếu có hai mô đun không phải là nối nội dung, cũng không nối công cộng, không nối ngoài không nối điều khiển và nếu chúng tham chiếu tới cùng cấu trúc dữ liệu không toàn cục, thì mối quan hệ giữa hai mô đun này được gọi là nối dấu. Mặc dầu nối dấu là tương tự với nối công cộng, sự khác biệt là ở chỗ các mô đun của nối dấu tham chiếu tới cấu trúc dữ liệu không toàn cục, không tới cấu trúc dữ liệu toàn cục.

Mô đun "đánh giá" được nêu trong Hình 4-2-40 được dùng để cập nhật "thu nhập hàng năm," bằng việc dùng một bản ghi nhân viên được nhập khẩu vào mô đun A như một đối, cũng như dữ liệu trên "vị trí công tác," "năng suất lao động" và "số năm công tác" của một nhân viên.

Hình 4-2-40 Nối dấu (ví dụ)



Để một mô đun thực hiện chức năng "đánh giá", ít nhất phải cần tới bốn khoản mục dữ liệu. Bởi vì toàn thể một bản ghi nhân viên vận hành như một đối, nên các khoản mục không cần tới khác bên cạnh bốn khoản mục này cũng được truyền. Do đó, nếu có bất kì thay đổi nào được thực hiện trong những khoản mục khác hơn bốn khoản mục này, thì mô đun chịu trách nhiệm thực hiện chức năng "đánh giá" phải được dịch lại. Hơn nữa, khi các khoản mục không cần tới có thể được tham chiếu tới, thì có khả năng là chúng bị cập nhật lầm. Mối quan hệ giữa mô đun "đánh giá" và tất cả các mô đun khác tham chiếu tới những khoản mục đặc biệt trong bản ghi nhân viên, được gọi là nối dấu.

Nối dấu làm cho chúng ta có khả năng giải quyết các vấn đề sau của nối công cộng:

- Thiếu tính dễ đọc và dễ hiểu
- Không phù hợp để được dùng cho các chủ đích đa dạng
- Phụ thuộc tên

Mặc dầu có thể làm giảm tải việc truy nhập dữ liệu điều khiển, nó không thể xoá hoàn toàn tải việc này. Đồng thời nó còn tồn tại những vấn đề như:

- Sự phụ thuộc không mong muốn giữa các mô đun
- Tạo ra cấu trúc câm
- Quá phơi bày dữ liệu

⑥ Nối dữ liệu

Để cải tiến các điều kiện liên quan tới mô đun thẩm định "đánh giá" vấn đề, mô đun chịu trách nhiệm "đánh giá" phải được thiết kế như một mô đun không biết gì về bản ghi nhân viên. Mô đun này cần ba khoản mục, "vị trí công tác," "năng suất lao động" và "số năm công tác." Nó cho ra một khoản mục "thu nhập hàng năm" dựa trên dữ liệu được chứa trong ba khoản mục này. Nếu chúng ta có thể định nghĩa bốn khoản mục này như các đối, chúng ta có thể tạo ra một mô đun không biết gì về bản ghi nhân viên.

GỌI đánh giá (bản ghi nhân viên)

↓

GỌI đánh giá (vị trí công tác, năng suất lao động, số năm công tác, thu nhập hàng

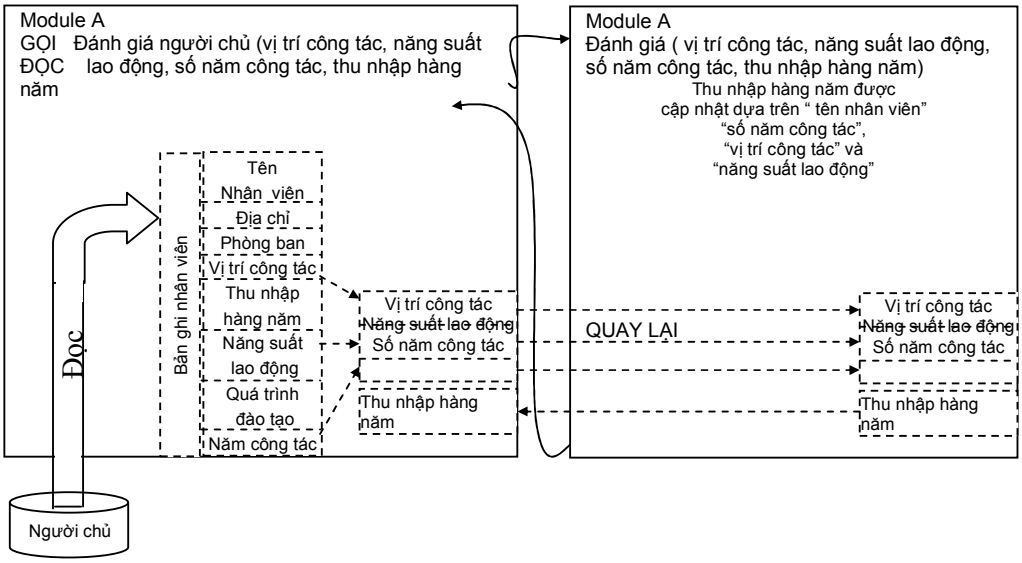
năm)

Giả định rằng mô đun trên mới được tạo ra và các yêu cầu sau được đáp ứng, mối quan hệ giữa các mô đun này được gọi là nối dữ liệu:

- Trạng thái của việc nối mô đun không phải là nối nội dung, công cộng, ngoài, điều khiển hay dấu..
- Hai mô đun có quan hệ trực tiếp với nhau.
- Tất cả các giao diện giữa các mô đun đều có cùng kiểu dữ liệu.

Nối dữ liệu là mức nối yếu nhất. (Xem Hình 4-2-41.)

Hình 4-2-41 Nối dữ liệu



(3) Tiêu chí cho kích cỡ phân hoạch

Kích cỡ thích hợp cho mô đun nói chung là 40 tới 50 câu lệnh dưới dạng số các lệnh thực hiện trong ngôn ngữ cấp cao. Nếu kích cỡ nhỏ hơn số này, thì người đọc chương trình thường nhảy từ mô đun này sang mô đun kia và luồng tư duy của người đó bị ngắt lại.

Nếu kích cỡ lớn hơn nhiều, tức là, nếu nó trên 100 câu lệnh, thì các giao diện được xác định rõ ràng sẽ không có sẵn và có quá nhiều thứ phải được xét tới.

Tiêu chí cho kích cỡ phân hoạch là như sau:

- Với chương trình cỡ nhỏ, có cấu trúc tốt (số các lệnh thực hiện được là 200 tới 300 câu lệnh): Trung bình là 30 câu lệnh
- Với chương trình cỡ vừa (số các lệnh thực hiện được là 2,000 tới 3,000): Trung bình 40 tới 50 câu lệnh
- Với chương trình cỡ lớn (số các lệnh thực hiện được là 10,000 hay hơn): Trung bình 100 tới 150 câu lệnh
- Với mô đun có độ bền thông tin (đây là trường hợp ngoại lệ): Trung bình 40 tới 50 câu lệnh cho một điểm vào

Những con số trên là trung bình và nên được dùng như hướng dẫn đơn thuần. Nếu các mô đun được phân hoạch hay được tổ hợp mà quá tuân theo các con số này, thì mức độ độc lập mô đun sẽ giảm đi. Phải thật chú trọng vào vấn đề này.

(4) Tạo ra và dùng lại các bộ phận

Để xem xét mô đun như các bộ phận và dùng lại chúng, chúng ta nên ý thức nhiều hơn tới tính độc lập của mô đun. Một mô đun với các chức năng được xác định mơ hồ, mô đun này phụ thuộc quá nhiều vào mô đun kia, mô đun này ảnh hưởng quá nhiều tới mô đun kia hay mô đun phụ thuộc vào cấu trúc dữ liệu đặc biệt thì không thể được dùng lại.
Mục tiêu của thiết kế có cấu trúc là để tạo ra chỉ các mô đun có độ bền chức năng hay thông tin. Nếu chỉ những mô đun như vậy mới có thể được tạo ra, thì tính độc lập mô đun có thể được đảm bảo, tỉ lệ lỗi có thể được giảm đi, tính mở rộng có thể được nâng cao, và xác suất dùng lại có thể được tăng lên.

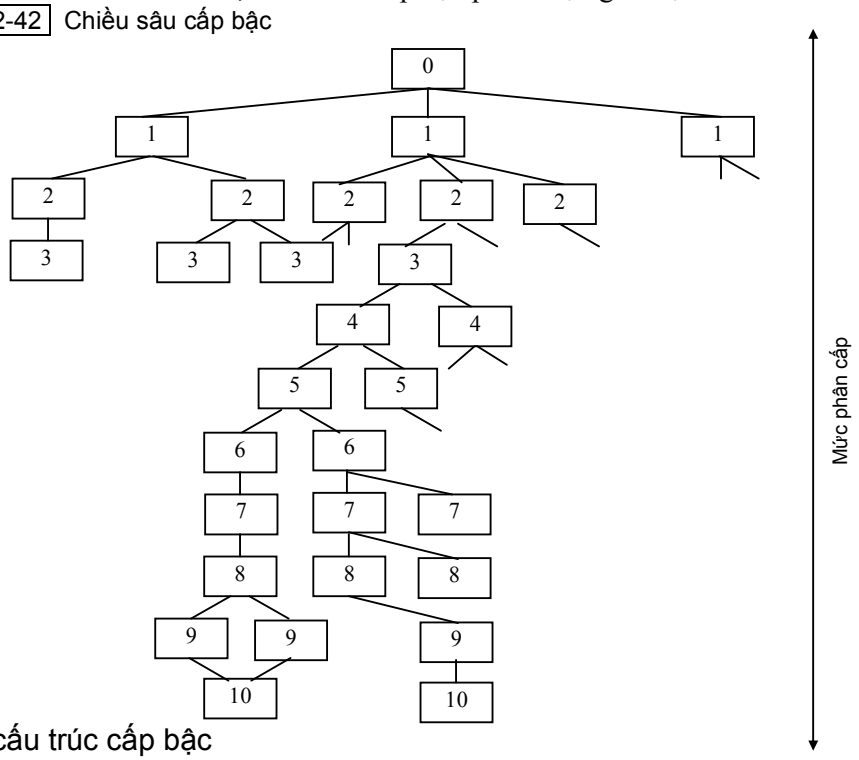
4.2.4 Phân hoạch chương trình

(1) Những điểm quan trọng cần xem xét liên quan tới số các phân hoạch và chiều sâu cấp bậc

① Chiều sâu cấp bậc

Nếu cấp bậc là quá sâu, thì cấu trúc logic trở nên khó hiểu. Mặc dầu kích cỡ của chương trình là nhân tố cần xem xét, chiều sâu cấp bậc phải được giới hạn vào mười mức hay ít hơn.

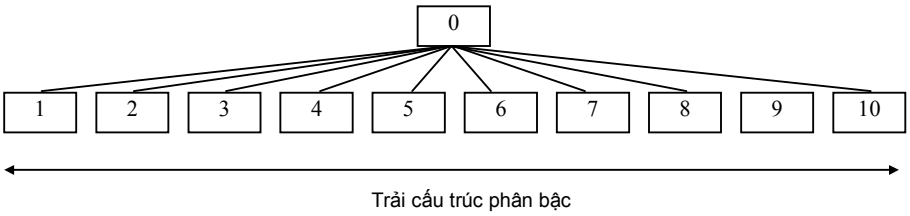
Hình 4-2-42 Chiều sâu cấp bậc



② Trải cấu trúc cấp bậc

Nếu cấp bậc của các mô đun trải ra quá nhiều, thì luồng chương trình trở thành khó hiểu. Số các mô đun cấp dưới mà một mô đun nào đó có thể trực tiếp gọi đến nên được giới hạn quãng mười hay ít hơn.

Hình 4-2-43 Trải cấu trúc cấp bậc



4.3 Tạo ra đặc tả mô đun và đặc tả kiểm thử

4.3.1 Tạo ra đặc tả mô đun

Sau khi chương trình được phân hoạch thành các mô đun, các chi tiết về xử lí được từng mô đun thực hiện sẽ được xác định (thiết kế logic mô đun).
Công việc viết mã phải được thực hiện có chú ý tới các chi tiết tỉ mỉ sao cho không bỏ sót chức năng.

(1) Những điểm quan trọng cần xét khi tạo ra các đặc tả mô đun

- Tạo ra cấu trúc đơn giản, dễ hiểu
- Tính tới việc dễ gỡ lỗi và bảo trì
- Tôn trọng chuẩn tạo ra chương trình (chuẩn viết mã)

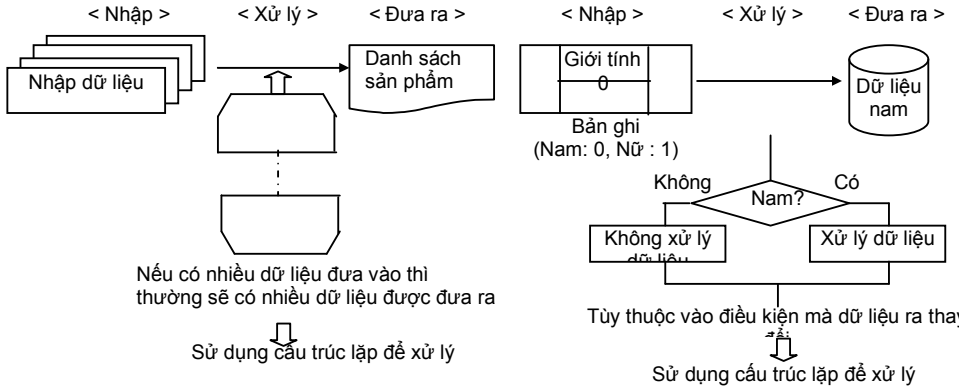
(2) Thủ tục để tạo ra các đặc tả mô đun

Các đặc tả mô đun được tạo ra theo thủ tục nêu sau đây.

① Phân tích cấu trúc dữ liệu

Bởi vì cấu trúc dữ liệu có mối quan hệ chặt chẽ với cấu trúc logic của mô đun, nên cấu trúc của dữ liệu vào và ra phải được phân tích, và mối quan hệ giữa dữ liệu vào và ra cũng như các khoản mục được sinh ra qua việc xử lí phải được nhận diện và liệt kê.

Hình 4-3-1 Đưa vào, xử lí và đưa ra



Nếu có nhiều khoản mục dữ liệu, thì việc xử lí dữ liệu được thực hiện bằng việc dùng một cấu trúc lặp lại. Nếu dữ liệu đưa ra thay đổi, tuỳ theo hoàn cảnh, việc xử lí dữ liệu được thực hiện bằng việc dùng cấu trúc lựa chọn.

② Phân tầng cấu trúc dữ liệu

Với việc ghi nhớ ý nghĩa của từng khoản mục dữ liệu, dữ liệu được tổ chức và tổ hợp lại, và cấu trúc dữ liệu được tạo ra. Mối quan hệ giữa dữ liệu vào và ra được làm sáng tỏ, và dữ liệu vào được liên kết với dữ liệu ra.

③ Xây dựng logic (thuật toán)

Việc chuyển đổi dữ liệu vào thành dữ liệu ra là nhiệm vụ cần thực hiện. Các điều kiện đưa ra được xác định khi xem xét tới cấu trúc của dữ liệu ra. Các điều kiện đưa ra đã xác định được dùng để làm rõ các chi tiết và để xây dựng thuật toán.
Logic này được viết thành tài liệu bằng việc dùng giả mã.

Hình 4-3-2 Giả mã

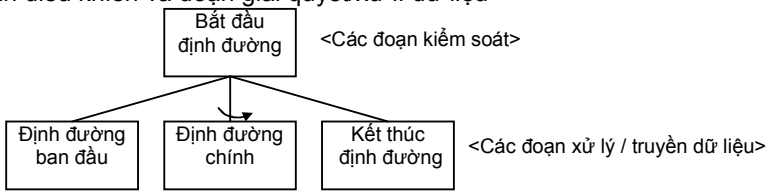
```
Đọc bản ghi đầu tiên
DO WHILE trong bản ghi
  IF lớp = trong kho THEN
     Xử lý trong kho
  ELSE
     IF lớp = ngoài kho THEN
        Xử lý ngoài kho
     ELSE
        Xử lý lỗi
     ENDIF
  ENDIF
  Đọc bản ghi
 ENDDO
```

(3) Đoạn và câu lệnh

Mô đun bao gồm các đoạn nhất quán về mặt logic. Kích cỡ của đoạn là từ 10 tới 50 câu lệnh (câu, lệnh).
Đoạn trong một mô đun được phân loại thành hai kiểu được nêu dưới đây:

- Đoạn điều khiển (đoạn mức cao) điều khiển tất cả các đoạn
- Đoạn giải quyết/xử lí dữ liệu (đoạn mức thấp) thực hiện nhiệm vụ xử lí dữ liệu

Hình 4-3-3 Đoạn điều khiển và đoạn giải quyết/xử lí dữ liệu



Nếu các điều kiện xử lí và chi tiết xử lí là như nhau, thì chúng nên được tổ hợp vào một đoạn.

4.3.2 Tạo ra đặc tả kiểm thử

Kiểm thử đơn vị là kiểm thử được tiến hành trong giai đoạn thiết kế chương trình. Chủ định của phép kiểm thử này là để kiểm chứng cấu trúc logic của mô đun đã viết mã và giao diện giữa các mô đun.

(1) Các kiểu kiểm thử

Kiểm thử đơn vị đầu tiên được tiến hành để kiểm chứng rằng cấu trúc logic của các mô đun được tạo ra theo đúng đặc tả mô đun. Kĩ thuật được dùng trong kiểm thử này là kiểm thử hộp trắng để loại bỏ đi các lỗi và phòng ngừa chúng không cho truyền qua các kiểm thử kế tiếp.
Kiểm thử tích hợp mô đun tiếp đó được tiến hành để kiểm tra giao diện giữa các mô đun. Đặc biệt, các mô đun được nối lại để xác nhận rằng các tham biến vào và ra được xác định trong

đặc tả mô đun, có thể được truyền đúng. Kĩ thuật được dùng trong kiểm thử này là kiểm thử hộp đen. Việc kiểm thử tính nối mô đun được chia thành kiểm thử trên xuống, được tiến hành bằng việc nối liên tiếp các mô đun cấp cao với mô đun cấp thấp, và kiểm thử dưới lên, được tiến hành bằng việc nối liên tiếp các mô đun cấp thấp với mô đun cấp cao. Mỗi phương pháp đều có ưu điểm và nhược điểm. Nên chọn một trong các phương pháp này khi xét tới kích cỡ của chương trình và tài nguyên sẵn có.

(2) Những điểm quan trọng cần xét khi thiết kế trường hợp kiểm thử

Nói chung, trường hợp kiểm thử không chỉ có nghĩa là dữ liệu kiểm thử mà còn là kế hoạch kiểm thử và các tài liệu khác. Trước khi tiến hành các kiểm thử tại từng mức, dữ liệu kiểm thử nên được chuẩn bị có xét tới các kĩ thuật kiểm thử được chấp thuận. Ngoài dữ liệu thông thường, dữ liệu sai cũng phải được chuẩn bị. Các kết quả đưa ra có thể dự kiến trước dựa trên dữ liệu kiểm thử cũng phải được chuẩn bị. Nếu phép kiểm thử được tiến hành có chú ý tới các chi tiết, thì chất lượng của chương trình có thể được cải tiến, nhưng sẽ cần tới nhiều công việc. Bởi vì năng suất là tỉ lệ nghịch với mức bao phủ, nên cần thiết kế dữ liệu kiểm thử với sự chú tâm nào đó để giữ cho chúng được cân bằng.

4.4 Tạo ra tài liệu thiết kế chương trình

Tài liệu thiết kế chương trình được tạo ra dựa trên nhiều dữ liệu và tài liệu đã được chuẩn bị trong giai đoạn thiết kế chương trình. Trong giai đoạn thiết kế chương trình, chương trình mà đã được xây dựng bằng thiết kế trong được phân hoạch thành các mô đun, và cấu trúc logic của mô đun được xác định. Do đó khi làm việc thiết kế chương trình, không được phân hoạch các mô đun nữa hay không được thiết kế cấu trúc logic của chúng theo cách có thể làm ảnh hưởng tới các chức năng của chương trình. Các chức năng này đã được thiết kế trong giai đoạn thiết kế trong để đáp ứng những yêu cầu đặc biệt.
Trong giai đoạn lập trình tiếp theo, công việc viết mã được thực hiện theo đúng tài liệu thiết kế chương trình. Có trường hợp người làm hợp đồng được đặt làm công việc thiết kế chương trình và các tiến trình kế tiếp. Theo hướng đó, tài liệu thiết kế chương trình là rất quan trọng, nó ảnh hưởng rất lớn tới thiết kế chương trình và các tiến trình tiếp theo. Do đó, việc kiểm điểm thiết kế phải được tiến hành một cách thấu đáo, dựa trên tài liệu thiết kế chương trình để rút ra và sửa mọi khiếm khuyết.
Hình 4-4-1 đưa ra các khoản mục có trong tài liệu thiết kế chương trình.

Hình 4-4-1 Tài liệu thiết kế chương trình

4.4.1 Tạo ra tài liệu thiết kế chương trình và nội dung

(1) Chính sách thiết kế chương trình

Chương trình được thiết kế dựa trên chính sách thiết kế chương trình sau đây:

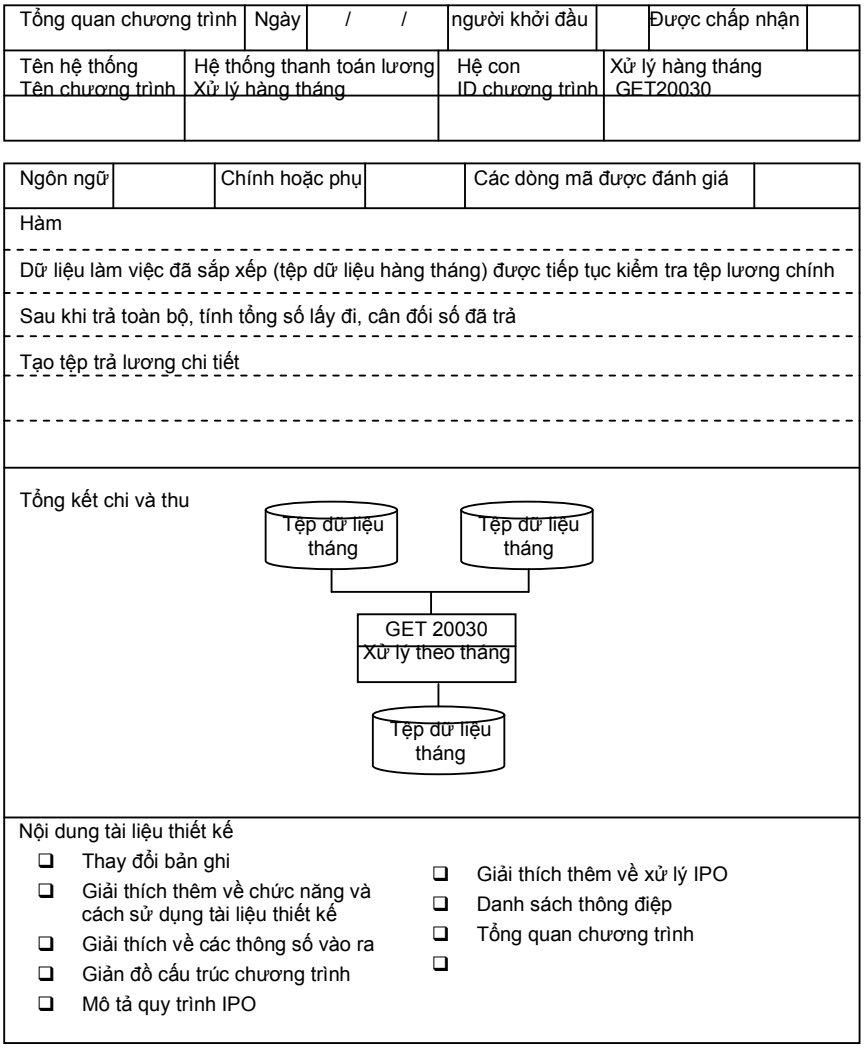
- Chính sách thiết kế
  Trước khi bắt đầu thiết kế chương trình, phải mô tả kĩ thuật thiết kế được chấp thuận. Thường thiết kế có cấu trúc hay được dùng. Nếu chọn kĩ thuật thiết kế khác, thì phải mô tả lí do cho việc chấp thuận này.
- Kĩ thuật làm tài liệu
  HIPO, DFD, lưu đồ, sơ đồ bọt, v.v.., được tham chiếu tới như kĩ thuật làm tài liệu. Phải mô tả các kĩ thuật được chấp thuận và cách chúng được chấp thuận.
- Nhiệm vụ thiết kế
  Mô tả nhiệm vụ thiết kế.

- Các vấn đề khác
  Phải mô tả các bản ghi thay đổi, chi tiết các cuộc kiểm điểm thiết kế và các vấn đề khác cần được xác định trước khi bắt đầu thiết kế chương trình.

(2) Bản tổng quan chương trình

Biểu đồ nêu ra cấu trúc chương trình (kể cả kiểu dữ liệu vào và ra) cần được đính với bản tổng quan chương trình. (Xem Hình 4-4-2.)

Hình 4-4-2 Bản tổng quan chương trình



(3) Biểu đồ cấu hình chương trình

Biểu đồ cấu hình chương trình và danh sách các giao diện giữa các mô đun cần được đính kèm. (Xem Hình 4-2-3.) Biểu đồ các nội dung được tạo ra bằng việc dùng kĩ thuật HIPO có thể được dùng như biểu đồ cấu hình mô đun.

(4) Chi tiết xử lí

Tài liệu mô tả chi tiết xử lí cần được từng mô đun thực hiện được đính kèm với các chi tiết xử lí. Biểu đồ IPO, v.v.. cũng có ích.

(5) Đặc tả kiểm thử

Bản kế hoạch kiểm thử mô tả nhân viên kiểm thử, khoản mục kiểm thử và phương pháp kiểm chứng cần được chuẩn bị để tiến hành các kiểm thử chương trình có hiệu quả.

(6) Mô tả các khoản mục dữ liệu

Màn hình được dùng để chạy chương trình và dữ liệu vào/ra được đính vào mô tả các khoản mục dữ liệu. Cũng có thể dùng màn hình và dữ liệu vào/ra do thiết kế ngoài và trong cung cấp. Những tài liệu sau đây được đính kèm với mô tả về các khoản mục dữ liệu:

- Các mẫu đưa vào và các báo cáo đưa ra
- Tài liệu thiết kế màn hình
- Tài liệu thiết kế tệp
- Đặc tả bảng
- Các tài liệu khác

4.4.2 Những điểm cần lưu ý khi tạo ra tài liệu thiết kế chương trình

Những điểm sau đây nên được xét tới khi tạo ra tài liệu thiết kế chương trình:

<Những điểm quan trọng cần xem xét>

- Các chức năng được mô tả trong tài liệu thiết kế trong phải được mô tả trong tài liệu thiết kế chương trình mà không bỏ sót.
- Tất cả các dữ liệu vào và ra phải được mô tả rõ ràng.
- Tất cả các qui tắc giải quyết lỗi phải được mô tả rõ ràng.
- Tiêu chí cho việc chuẩn bị tài liệu phải được tôn trọng và cần tránh những các biểu thức lạc lõng.

4.4.3 Họp kiểm điểm thiết kế

Họp kiểm điểm thiết kế được thực hiện khi công việc thiết kế chương trình được hoàn tất, như trong trường hợp của giai đoạn thiết kế trong. (Tham khảo chi tiết ở Chương 3.)

(1) Tài liệu cần được kiểm điểm

Tài liệu cần được kiểm điểm trong giai đoạn thiết kế chương trình là:

- Tài liệu thiết kế chương trình
- Biểu đồ cấu trúc chương trình
- Đặc tả chương trình
- Đặc tả mô đun
- Tài liệu thiết kế trường hợp kiểm thử
- Các tài liệu khác

(2) Nhân sự kiểm điểm

Các nhân sự nêu dưới đây giữ vai trò trung tâm trong việc thực hiện buổi kiểm điểm thiết kế:

- Những người thiết kế có cùng mức kĩ năng kĩ thuật như người trực tiếp chịu trách nhiệm thiết kế chương trình
- Các nhân sự có liên quan tới tiến trình thiết kế

Cấp trên của người thiết kế và của các nhân sự này không cần tham dự vào cuộc họp kiểm điểm thiết kế.

Bài tập

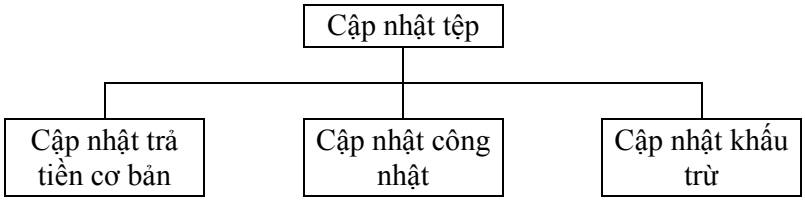
Q1 **Nhận xét nào là <u>không thích hợp</u> liên quan tới việc phân hoạch mô đun trong giai đoạn thiết kế chương trình?**

a. Số các mô đun cấp dưới mà một mô đun có thể gọi tới phải được giới hạn.
b. Mô đun phải được thiết kế sao cho nó chứa một số đúng các bước.
c. Trong thiết kế cấu trúc cấp bậc mô đun này gọi tới mô đun kia, phải để ý tới việc giữ chiều sâu trong giới hạn xác định.
d. Giao diện giữa các mô đun phải được làm đơn giản hoá.
e. Nên đưa vào các chú thích đúng để làm cho dễ hiểu logic bên trong mô đun.

Q2 **Khi một chương trình đọc dữ liệu, lựa ra chỉ dữ liệu số và cho hiện ra giá trị trung bình, được lấy ra làm phân hoạch STS, thì các chức năng được sắp xếp vào các loại bể chứa, nguồn và biến đổi. Hãy lựa ra tổ hợp đúng từ những tổ hợp được nêu dưới đây.**

	Chức năng			
	Vào dữ liệu	Lựa số	Tính giá trị trung bình	Biểu thị kết quả
a	Bể chứa	Bể chứa	Biến đổi	Biến đổi
b	Bể chứa	Nguồn	Biến đổi	Biến đổi
c	Nguồn	Nguồn	Biến đổi	Bể chứa
d	Nguồn	Biến đổi	Biến đổi	Bể chứa
e	Biến đổi	Đưa ra	Đưa ra	Nguồn

Q3 **Có một chương trình để chấp nhận cập nhật trả tiền cơ bản, cập nhật tiền công nhật và cập nhật khấu trừ và cập nhật tệp tính lương. Chương trình này được phân hoạch thành các mô đun, như được nêu dưới đây. Phương pháp phân hoạch mô đun nào được dùng?**



a. Phương pháp phân hoạch STS b. Phương pháp Jackson
c. Phương pháp phân hoạch giao tác d. Phương pháp Warnier

Q4 **Bạn nên dùng phương pháp nào để chuyển biểu đồ luồng dữ liệu được phân tích có cấu trúc tạo ra thành sơ đồ cấu trúc được dùng cho thiết kế có cấu trúc?**

a. Phương pháp KJ b. Phương pháp OMT c. Phương pháp Jackson
d. Phương pháp phân hoạch giao tác

Q5 **Kĩ thuật nào sau đây là kĩ thuật phân hoạch mô đun hướng cấu trúc?**

a. Phương pháp phân hoạch chức năng chung
b. Phương pháp phân hoạch nguồn/biến đổi/bể chứa (phương pháp phân hoạch STS)
c. Phương pháp Jackson
d. Phương pháp phân hoạch giao tác (Phương pháp phân hoạch TR)

Q6 **Lưu ý nào được nêu dưới đây mô tả sát nhất cho phương pháp Warnier được dùng để tạo ra thiết kế có cấu trúc của chương trình?**

a. Biểu đồ cấu trúc của dữ liệu vào và ra được vẽ với chú ý chính dồn vào cấu trúc dữ liệu. Biểu đồ cấu trúc chương trình được chuẩn bị dựa trên biểu đồ cấu trúc dữ liệu vào/ra.
b. Các chức năng trong luồng dữ liệu được gộp nhóm vào trong các loại nguồn, biến đổi và bể chứa với chú ý chính được dồn vào luồng dữ liệu cần giải quyết.
c. Phần mềm được coi như một tuyển tập các dữ liệu và qui trình. Tính độc lập mô đun được tăng lên bằng cách bao bọc những dữ liệu và qui trình này.
d. Với chú ý chính được dồn vào cấu trúc điều khiển của chương trình, logic chương trình được thiết kế dựa trên luồng điều khiển, biểu lộ mối quan hệ gọi nhau.

Q7 **Tính nối mô đun là cách đo sự độc lập mô đun. Tính nối mô đun càng yếu, thì mức độ độc lập mô đun càng trở nên cao hơn. Kiểu nối mô đun nào được nêu dưới đây có tính nối mạnh nhất?**

a. Nối công cộng b. Nối dấu c. Nối dữ liệu d. Nối dội dung

Q8 **Khi chương trình sau được thực hiện, bạn thu được kết quả nào trong số được nêu dưới đây? x (đối hình thức) là lời gọi theo giá trị còn y là lời gọi theo tham chiếu.**

```
Main program          Subprogram  sub (x, y)
  a=3 ;                 x=x+y ;
  b=2 ;                 y=x+y ;
  sub (a, b) ;          return ;
```

a. a=3, b=2 b. a=3, b=7 c. a=5, b=2 d. a=5, b=7

5 Thực hiện chương trình

Mục đích của chương

Trong giai đoạn thực hiện chương trình, hệ thống thông tin được thiết kế theo các thủ tục đã được mô tả, sẽ được xây dựng nên.
Chương này mô tả những điểm quan trọng cần chú ý khi làm việc lập trình (viết mã), các phương pháp kiểm thử và các công cụ phát triển khác nhau mà chúng ta có thể dùng.

① Hiểu mô thức lập trình và phong cách lập trình
② Hiểu các kiểu khác nhau về kiểm thử, phương pháp kiểm thử và thủ tục kiểm thử
③ Hiểu các kiểu và đặc trưng của các công cụ phát triển khác nhau mà bạn có thể dùng cho việc lập trình và kiểm thử

Giới thiệu

Cài đặt chương trình là cho một dạng cụ thể của chương trình được thiết kế về mặt logic. Đặc biệt nó bao gồm tiến trình tạo ra chương trình dựa trên nội dung của thiết kế chương trình, và tiến trình thực hiện những phép kiểm thử đa dạng trước khi cho chạy chương trình như một hệ thống.

5.1 Lập trình

Lập trình là mô tả (viết mã) trong một ngôn ngữ lập trình cho một thủ tục (thuật toán) được xác định bởi thiết kế chương trình.
Mỗi ngôn ngữ lập trình đều gán nghĩa riêng của nó cho các lệnh và cú pháp để thống nhất chúng, và do vậy mô tả cho các thuật toán theo những cách khác nhau. Do đó, cần có một chuẩn chung (mô thức lập trình) để xét tới những đặc trưng của từng ngôn ngữ lập trình.
Bởi vì công việc lập trình được thực hiện bằng nhóm người, nên phong cách lập trình rõ ràng là cần thiết để đảm bảo sự nhất quán bên trong hệ thống.

5.1.1 Mô thức lập trình

Mỗi ngôn ngữ lập trình được dùng đều có mô thức riêng của nó. Trong khi làm việc lập trình, cần hiểu từng mô thức lập trình riêng.
Bởi vì mô thức phụ thuộc vào từng ngôn ngữ lập trình, nên nó phải được nghiên cứu cho từng ngôn ngữ một. Chúng ta phân lớp nó một cách đại thể thành bốn kiểu tương ứng với việc phân lớp về ngôn ngữ lập trình:

- Mô thức lập trình thủ tục
- Mô thức lập trình logic
- Mô thức lập trình hàm
- Mô thức lập trình hướng đối tượng

(1) Mô thức lập trình thủ tục

Lập trình thủ tục là mô thức của các ngôn ngữ lập trình thủ tục mô tả cho giải pháp vấn đề dưới dạng một chuỗi các thủ tục. C và COBOL là đại biểu cho các ngôn ngữ lập trình thủ tục.
Một đặc trưng của ngôn ngữ lập trình này là lập trình có cấu trúc. Khái niệm về lập trình có cấu trúc dự định diễn đạt mọi thuật toán bằng việc dùng ba cấu trúc điều khiển cơ sở (tuần tự, tuyển chọn và lặp). Bằng việc dùng lược đồ lập trình này, chúng ta có thể tối thiểu việc dùng câu lệnh go to thường gây ra sự suy giảm trong công tác bảo trì.

(2) Mô thức lập trình logic

Lập trình logic là mô thức của ngôn ngữ lập trình logic mô tả cho giải pháp vấn đề dưới dạng

các khai báo logic. Prolog là đại diện cho ngôn ngữ lập trình logic.
Đặc trưng của ngôn ngữ này là qui tắc giải dựa trên tam đoạn luận. Trong trường hợp của Prolog, ba cú pháp được dùng: qui tắc, sự kiện và truy vấn. Việc sánh mẫu (thống nhất), việc tìm tự động (lần ngược) v.v.. được dùng làm các cấu trúc cơ sở.

(3) Mô thức lập trình hàm

Lập trình hàm là mô thức của ngôn ngữ lập trình hàm mô tả cho giải pháp vấn đề dưới dạng các khai báo hàm. LISP là đại diện cho ngôn ngữ lập trình hàm.
Đặc trưng của ngôn ngữ này là việc dùng một cấu trúc danh sách. Bởi vì các đối tượng (dữ liệu) được xử lí đều giải quyết với cấu trúc danh sách, nên nhiều hàm được cung cấp để xử lí danh sách. Để mô tả và định nghĩa các hàm, một hệ thống ngôn ngữ trừu tượng cao gọi là tính toán lambda được sử dụng.

(4) Mô thức lập trình hướng đối tượng

Lập trình hướng đối tượng là mô thức của ngôn ngữ lập trình hướng đối tượng dùng đối tượng có dữ liệu và thủ tục (hành vi) được bao bọc lại. Smalltalk và Java là các ngôn ngữ lập trình hướng đối tượng đại diện.
Các đặc trưng của ngôn ngữ lập trình này là:

- Chức năng bao bọc
  Chức năng này giải quyết với dữ liệu (thuộc tính) và thủ tục (động pháp) như một thực thể.
- Chức năng xây dựng thể nghiệm
  Chức năng làm thể nghiệm cho một lớp trừu tượng
- Chức năng kế thừa
  Chức năng kế thừa các tính chất của một siêu lớp cho lớp con
- Chức năng truyền thông báo
  Chức năng truyền thông báo giữa các đối tượng.

5.1.2 Phong cách lập trình

Phong cách lập trình là cơ sở để dựa vào đó chương trình được tạo ra. Khi hệ thống trở nên lớn về qui mô, nhiều người phát triển tham gia vào công việc lập trình và do đó cần có một phong cách lập trình rõ ràng.
Phong cách lập trình được xác định từ các quan điểm sau:

- Tính rõ ràng
- Tính hiệu quả
- Tính bảo trì được

(1) Tính rõ ràng

Tính rõ ràng nghĩa là khả năng hiểu được chương trình. Để làm tăng mức độ rõ ràng, nói chung người ta thiết lập ra những qui tắc viết mã (chuẩn).
Qui tắc viết mã xác định các qui tắc cần tuân thủ khi viết mã: tụt lề, đặt tên biến và mô đun, lời chú thích v.v.. Bằng việc tuân theo các qui tắc này, ta có thể tạo ra một chương trình dễ

hiểu đối với những người khác. Công việc viết mã, được tiến hành tuân thủ theo các qui tắc này, dường như tốn nhiều thời gian hơn công việc viết mã được làm mà không có qui tắc nào. Việc viết mã được thực hiện theo các qui tắc này cuối cùng dẫn tới việc làm giảm thời gian dành cho kiểm điểm lại, tuy nhiên, lại cho phép tổng lượng thời gian phát triển chương trình được rút bớt.

(2) Tính hiệu quả

Tính hiệu quả nghĩa là dễ dàng tạo ra chương trình. Để làm tăng tính hiệu quả, những phần dư thừa của chương trình phải được loại bỏ tối đa. Tuy nhiên cũng nên lưu ý ở đây rằng tính rõ ràng của chương trình có thể được tăng thêm nhiều bằng những phần bổ sung vào chương trình. Chẳng hạn, mặc dầu lời chú thích là phần phụ thêm không liên quan tới việc thực hiện chương trình, nó vẫn đóng góp làm tăng mức độ rõ ràng của chương trình. Do đó, cần giữ tính rõ ràng và tính hiệu quả được cân bằng tốt.

(3) Tính bảo trì được

Tính bảo trì nghĩa là dễ sửa đổi chương trình. Người làm lập trình thường không sửa đổi chương trình. Việc sửa này do người khác thực hiện. Để làm cho công việc bảo trì được dễ dàng, điều quan trọng là cần tạo ra chương trình dễ hiểu.
Để nâng cao tính bảo trì được, bản thân chương trình phải được cấu trúc cao để ngăn cản việc sửa đổi thực hiện ở phần này có thể ảnh hưởng tới phần khác của chương trình.

5.1.3 Dùng bộ xử lí ngôn ngữ

Bộ xử lí ngôn ngữ là từ chung để nói tới các chương trình thực hiện các nhiệm vụ dịch và soạn thảo để làm cho chương trình đã viết chạy được.
Các bộ xử lí ngôn ngữ đại diện là:

- Bộ biên dịch, dịch chương trình viết trong ngôn ngữ cấp cao thành chương trình mã máy ngay lập tức.
- Bộ hợp dịch, dịch chương trình được viết trong hợp ngữ thành ngôn ngữ máy ngay lập tức.
- Bộ thông dịch, dịch và thực hiện các câu lệnh trong chương trình được viết trong ngôn ngữ cấp cao theo từng lệnh một.

Trong khi làm công việc lập trình, người ta phải sử dụng tới các đặc trưng của từng bộ xử lí ngôn ngữ này.
Các đặc trưng của bộ thông dịch là:

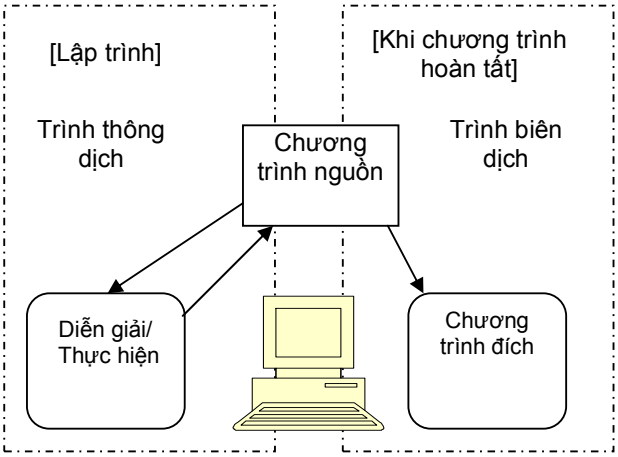
- Việc thực hiện lần lượt từng lệnh là có thể được.
- Chương trình có thể được cho chạy ngay cả khi chưa hoàn thành.

Với việc tận dụng các đặc trưng này, có thể thu được những ích lợi sau:

- Có thể kiểm chứng được hành vi của chương trình tại từng điểm nơi nó mới hoàn thành nửa chừng.
- Công việc gỡ lỗi được thực hiện dễ dàng (có thể dễ dàng thêm vào các lệnh gỡ lỗi).

Tốc độ thực hiện của chương trình được viết trong bộ thông dịch tuy vậy lại chậm hơn so với tốc độ của chương trình được biên dịch. Do đó trong việc lập trình thực tế, bộ thông dịch được dùng trước hết để hoàn thiện chương trình, rồi bộ biên dịch được dùng để tăng tốc việc thực hiện nó.

Hình 5-1-1 Dùng các bộ xử lí ngôn ngữ



Một bộ xử lí ngôn ngữ được dùng khá thông dụng khác là bộ tiền xử lí. Theo nghĩa hẹp, bộ tiền xử lí được dùng để thực hiện các mở rộng macro trong câu lệnh chương trình hay để nhập khẩu các tệp. Trong việc lập trình thực tế, bạn nên nhận biết rằng bộ tiền xử lí chạy để thực hiện những chức năng này. Theo nghĩa rộng, bộ tiền xử lí được dùng để chuyển một chương trình viết trong một ngôn ngữ cấp cao này thành một chương trình được viết trong một ngôn ngữ cấp cao khác. Do đó với việc dùng bộ tiền xử lí, trước hết bạn có thể xây dựng một chương trình bằng việc dùng ngôn ngữ lập trình x, rồi chuyển đổi nó sang định dạng của ngôn ngữ lập trình y và cho chạy chương trình đã chuyển đổi này. Phương pháp này dùng một bộ xử lí ngôn ngữ có liên kết với bộ xử lí ngôn ngữ khác, là có ích nếu trình biên dịch y được dùng rộng rãi hơn trình biên dịch x, hay nếu việc dùng hai bộ xử lí ngôn ngữ khác nhau có thể làm tăng tính hiệu quả tối ưu. Bởi vì yêu cầu tiên quyết cho kĩ thuật lập trình này là chuyển đổi trơn tru từ x sang y, nên x thường là một phiên bản mở rộng của y.

5.1.4 Môi trường lập trình

Bộ xử lí ngôn ngữ và các công cụ phát triển khác được đưa vào trong môi trường phát triển chương trình. Hệ thống lập trình được dùng nhiều trong những năm gần đây là IDE, có các bộ xử lí ngôn ngữ và công cụ phát triển được tích hợp trong một hệ thống.
Tương tự, các công cụ phát triển đã được thiết kế để hỗ trợ cho các kĩ thuật lập trình mới như lập trình web đang được phát triển.

(1) IDE (Môi trường phát triển tích hợp - integrated development environment)

IDE cho phép một loạt các nhiệm vụ lập trình từ soạn thảo chương trình nguồn cho tới việc biên dịch, và từ móc nối cho tới gỡ lỗi đều được thực hiện trong một môi trường liên tục. Công việc lập trình trước đây đã được thực hiện bằng việc dùng các công cụ khác nhau để thực hiện từng nhiệm vụ lập trình. Do đó luồng lập trình trơn tru bị ngắt quãng, gây ra giảm tính hiệu quả và năng suất. IDE cung cấp giải pháp cho những vấn đề này, cho phép tất cả các nhiêm vụ lập trình được thực hiện trong một môi trường tích hợp.

Các đặc trưng của IDE là như sau:
- Tương hợp với các ngôn ngữ lập trình thường được dùng (C, COBOL, v.v.)
- Được thiết kế để cho phép hàng loạt nhiệm vụ lập trình thực tế được thực hiện.
- Móc nối với hệ quản trị cơ sở dữ liệu

Các sản phẩm đại diện của IDE là:
- Visual Studio 6.0 (Microsoft): Môi trường phát triển Windows dùng VB, VC++ và các ngôn ngữ khác trong một gói.
- Delphi 5.0 (Borland): Hệ thống phát triển trực quan dùng trình biên dịch tốc độ cao
- Developer 2000 (Oracle): Môi trường phát triển cho các ứng dụng cơ sở dữ liệu dùng Oracle.

(2) Lập trình Web

Có hai cách tiếp cận được tính tới để tạo ra việc lập trình web bằng việc dùng môi trường web:
- Làm cho chương trình hiện thực hiện được trong môi trường web.
- Phát triển một chương trình mới với dự định cho chạy nó trong môi trường web.

Một công cụ phát triển thích hợp cho cách tiếp cận thứ nhất trên đây là VB-web và công cụ thích hợp cho cách tiếp cận thứ hai là FrontPage. Cả hai đều do Microsoft cung cấp.

Các chức năng cần cho công cụ phát triển web là:
- Chức năng xây dựng GUI đơn giản
- Chức năng thiết kế khung chương trình dùng thuật sĩ
- Dùng ASP (Active Server Page - trang nguồn phục vụ tích cực)
- Tương hợp với các ngôn ngữ qui ước

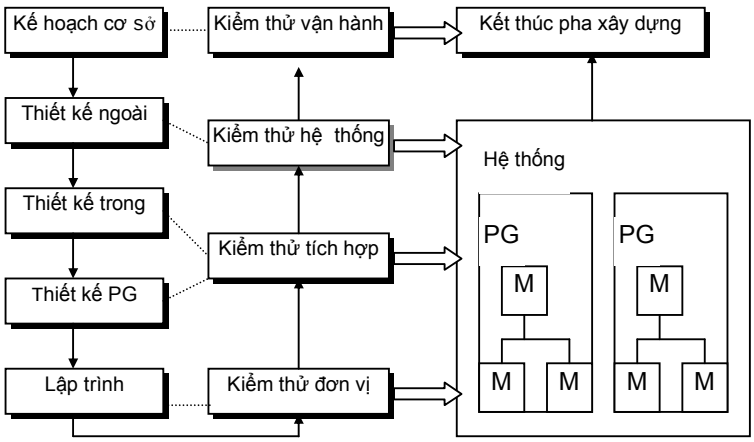
5.2 Kiểm thử

Người ta thường nói một chương trình mới được tạo ra chứa vài lỗi trong 100 dòng. Trong những dòng này, lỗi từ tiến trình lập trình và lỗi từ tiến trình thiết kế đều có cả. Nếu một chương trình chứa lỗi được dùng để vận hành một hệ thống trực tuyến, thì những hư hỏng nghiêm trọng phát sinh ra không chỉ ảnh hưởng tới công ti vận hành hệ thống đó, mà còn ảnh hưởng cả tới công chúng lớn bên ngoài.
Do đó, việc kiểm thử sản phẩm phải được tiến hành trước khi chuyển giao sản phẩm công nghiệp. Việc kiểm thử chương trình cũng phải được tiến hành theo một trình tự kiểm thử đặc biệt để kiểm chứng rằng chương trình và hệ thống mà nó điều khiển, có thể vận hành tương ứng với các đặc tả. Mặc dầu chúng ta không thể đảm bảo hoàn toàn loại bỏ hết lỗi trong chương trình, chúng vẫn có thể làm giảm số lỗi đó tới mức tối thiểu nhất nếu chúng ta kiểm thử chương trình theo cách chính xác, hiệu quả.

5.2.1 Tổng quan về kiểm thử

Trong mô hình thác đổ, kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống và kiểm thử vận hành được tiến hành theo thứ tự đó. Phía phát triển hệ thống thực hiện bước đầu trong tiến hành kiểm thử đơn vị, tích hợp và hệ thống, trong khi bộ phận người sử dụng chịu trách nhiệm tiến hành kiểm thử vận hành.

Hình 5-2-1 Tổng quan về kiểm thử



Khi bộ phận người dùng hoàn thành kiểm thử vận hành, thì chương trình được chính thức chuyển từ tổ chức hệ thống sang người dùng. Sau điều này, tổ chức người dùng nhận trách nhiệm quản lí chương trình.

5.2.2 Kiểm thử đơn vị

Kiểm thử đơn vị được tiến hành tại những giai đoạn sớm nhất trong pha kiểm thử. Mục tiêu kiểm thử là từng mô đun.

(1) Đại cương và mục đích của kiểm thử đơn vị

Kiểm thử đơn vị được tiến hành cho từng mô đun, đơn vị nhỏ nhất bên trong hệ thống đang xây dựng. Mục đích là kiểm chứng lại công việc đã được làm trong pha lập trình. Trong kiểm thử đơn vị thực tế, việc kiểm chứng được thực hiện để xem liệu các chức năng mô đun có tương ứng với đặc tả mô đun hay không.
Sau khi các mô đun đã được móc nối và tích hợp vào mức hệ thống chương trình, thì một số lớn công việc cần làm là loại bỏ lỗi. Để tránh điều này, lỗi trước hết phải được loại bỏ khỏi từng mô đun trong giai đoạn kiểm thử đơn vị, trước khi các mô đun được tích hợp lại.

(2) Phương pháp kiểm thử và thiết kế các trường hợp kiểm thử

① Phương pháp kiểm thử

Về nguyên tắc, kiểm thử hộp đen thường được tiến hành. Kiểm thử hộp trắng được tiến hành nếu cần.

- Kiểm thử hộp trắng (còn gọi là kiểm thử hộp trong): Chú ý chính được dồn vào cấu trúc bên trong.
- Kiểm thử hộp đen: Chú ý chính được dồn vào giao diện (cái vào và cái ra) giữa các mô đun.

② Thiết kế trường hợp kiểm thử

Trước khi kiểm thử chương trình, phải chuẩn bị các trường hợp kiểm thử (dữ liệu kiểm thử). Đó là một nhân tố quan trọng cần xét tới vì nó ảnh hưởng tới kết quả của việc kiểm thử, hay thậm chí còn xác định ra chất lượng của hệ thống.
Để thiết kế các trường hợp kiểm thử tối ưu, nếu có sẵn một hướng dẫn nào đó (tài liệu thiết kế trường hợp kiểm thử) thì sẽ rất có ích. Bằng cách tích luỹ các dữ liệu và xem lại tài liệu thiết kế trường hợp kiểm thử, tổ chức phát triển có thể lưu giữ cách làm để cải tiến chất lượng phần mềm, và dùng nó như phương tiện để truyền dữ liệu sang giai đoạn lập trình sau.

5.2.3 Kiểm thử tích hợp

Kiểm thử tích hợp được tiến hành sau khi kiểm thử đơn vị đã được hoàn tất. Chúng được dự định để kiểm chứng lại những công việc đã được tiến hành trong các pha thiết kế chương trình và thiết kế chương trình.

(1) Đại cương và mục đích của kiểm thử tích hợp

Kiểm thử tích hợp được tiến hành để kiểm chứng rằng nhiều mô đun có thể vận hành đúng khi được nối với các mô đun khác có liên quan. Trong khi tiến hành việc kiểm thử này, chú ý chính được dành cho giao diện giữa các mô đun (giao diện giữa các chương trình). Cho dù không tìm thấy vấn đề gì trong kiểm thử đơn vị, lỗi vẫn thường xuất hiện khi các mô đun được nối lại. Nếu lỗi xuất hiện trong kiểm thử tích hợp thì bạn phải quay lui trở về tiến trình trước và sửa vấn đề. Hãy nhớ rằng bạn cũng phải chữa lại tài liệu thiết kế.
Trước khi tiến hành kiểm thử tích hợp, phải định nghĩa rõ ràng các mô đun được nối theo thứ

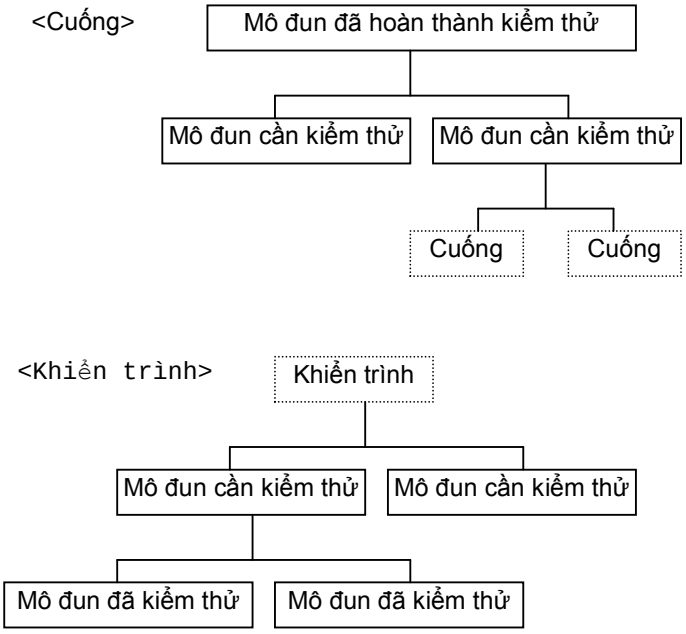
tự nào và khi nào.

(2) Cuống và khiển trình

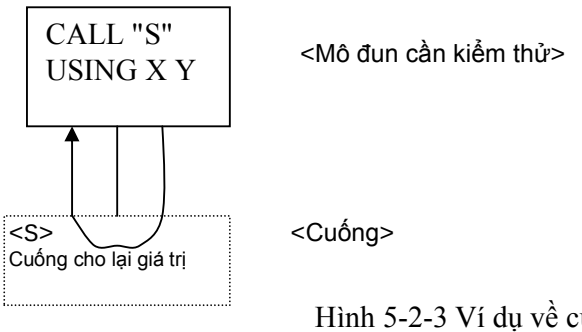
Trong giai đoạn lập trình của phát triển hệ thống, chương trình có cấu trúc phân cấp bao gồm nhiều mô đun. Do đó, việc mã hoá được thực hiện cho từng mô đun riêng, và các mô đun cấp cao hay cấp thấp được cần tới để kiểm chứng sự vận hành bình thường của các mô đun đã xây dựng. Trong kiểm thử thực tế, các mô đun câm được gọi là cuống hay khiển trình sẽ được dùng tới.

- Cuống (stub): Một chương trình dành cho kiểm thử để cung cấp các chức năng của mô đun mức thấp.
- Khiển trình (driver): Một chương trình dành cho kiểm thử để cung cấp các chức năng của các mô đun cấp cao

Hình 5-2-2 Cuống và khiển trình



Hình 5-2-3 nêu ra một ví dụ về cuống



Hình 5-2-3 Ví dụ về cuống.

(3) Kiểm thử tăng dần

Trong kiểm thử tăng dần, các mô đun đã hoàn tất kiểm thử sẽ được móc nối liên tiếp với các mô đun khác. Kiểm thử tăng dần được phân loại đại thể thành ba kiểm thử được nêu dưới đây:

- Kiểm thử trên xuống
- Kiểm thử dưới lên
- Kiểm thử tổ hợp (kiểm thử bánh mì kẹp thịt)

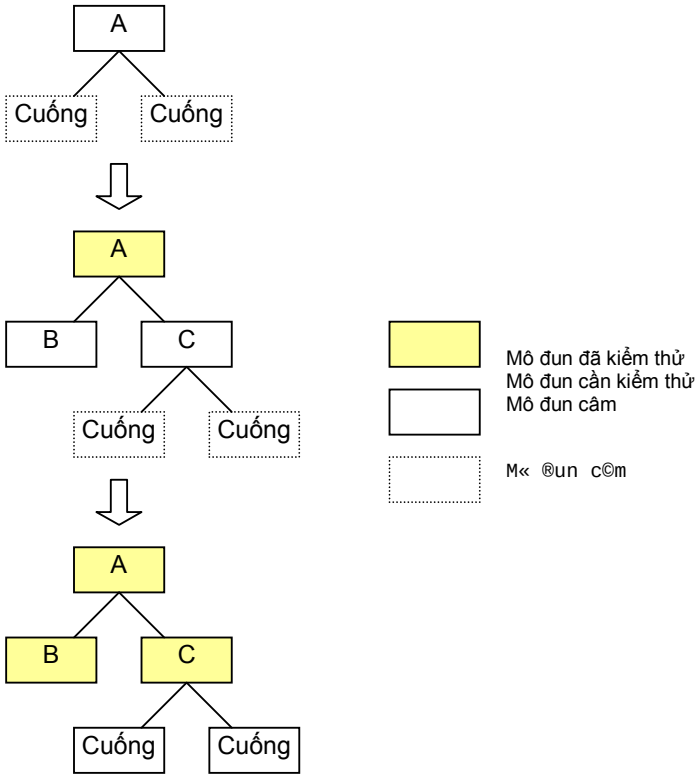
<Đặc trưng>

- Thích hợp cho kiểm thử chương trình kích cỡ lớn.
- Cần dùng các mô đun kiểm thử (mô đun câm) như các cuống và khiển trình thay cho các mô đun chưa hoàn thành.
- Kết quả của việc kiểm thử có thể thay đổi, tuỳ theo các mô đun được móc nối vào theo trình tự nào.
- Dễ theo dõi dấu vết lỗi về nguyên nhân.

① Kiểm thử trên xuống

Kiểm thử trên xuống được dùng để phát triển hệ thống theo thứ tự từ các mô đun cao tới mô đun thấp (được gọi là lập trình trên xuống).

5.2.3.1 Hình 5.54 Kiểm thử trên xuống



<Đặc trưng>

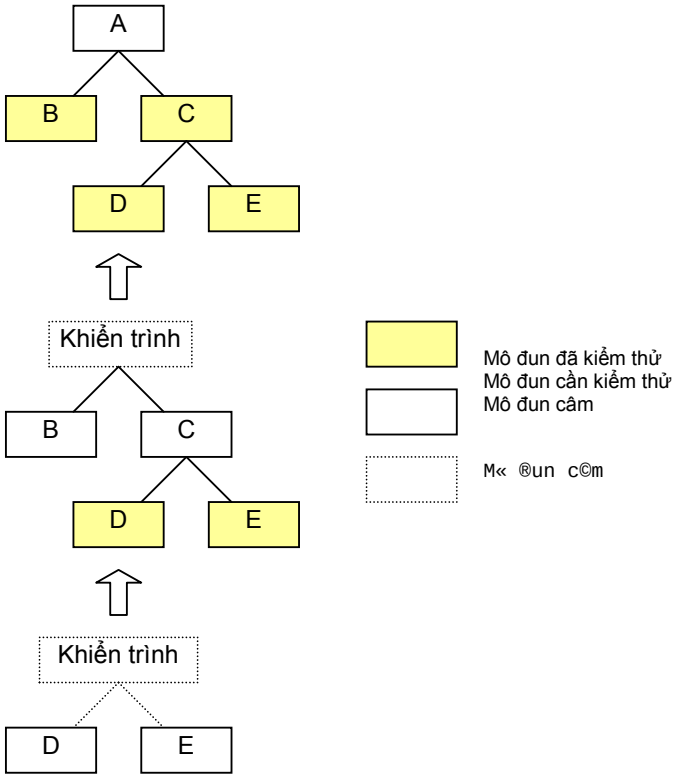
- Mô đun mức cao nhất (mô đun lõi hay mô đun có logic) trước hết được móc nối với mô đun cao nhất tiếp đó và tất cả các mô đun khác cũng được móc nối giống thế theo trình tự các mô đun từ mức cao tới thấp.

- Các mô đun quan trọng được kiểm thử thường xuyên hơn các mô đun kém quan trọng, do vậy làm tăng độ tin cậy của giao diện giữa các mô đun cấp cao.
- Tiền điều kiện cho việc dùng kiểm thử này là ở chỗ bản thân chương trình phải được tạo ra bằng việc dùng thiết kế có cấu trúc.
- Bởi vì mô đun cấp cao với một số nhỏ các chương trình là được phát triển trước, nên khó làm việc lập trình và tiến hành kiểm thử song song tại giai đoạn khởi đầu.
- Thích hợp cho việc phát triển hệ thống mới
- Xem như (chương trình) hệ thống kiểm thử, cần dùng tới cuống.

② Kiểm thử dưới lên

Kiểm thử dưới lên được dùng để phát triển hệ thống theo trình tự các mô đun mức thấp tới mức cao (được gọi là lập trình dưới lên).

Hình 5-2-5 Kiểm thử dưới lên



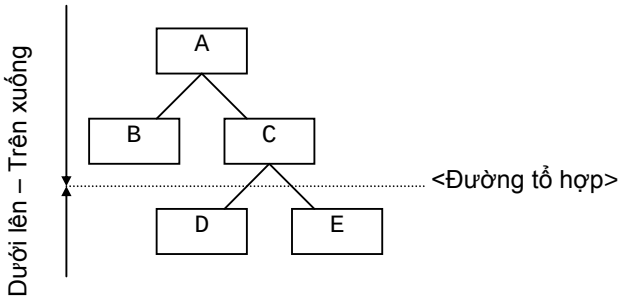
<Đặc trưng>

- Kiểm thử được tiến hành bằng việc móc nối các mô đun theo trình tự mô đun mức thấp tới cao.
- Khi kiểm thử tích hợp đã được hoàn tất, thì chương trình có thể được kiểm thử theo điều kiện vận hành thực tế.
- Bởi vì các mô đun mức thấp với một số lớn chương trình là được phát triển trước hết, nên có thể làm việc lập trình, và tiến hành kiểm thử song song tại giai đoạn khởi đầu.
- Xem như hệ thống (chương trình) kiểm thử, cần có khiển trình.
- Thích hợp cho việc phát triển một phiên bản sửa đổi của hệ thống hiện tại.

③ Kiểm thử tổ hợp (kiểm thử bánh mì kẹp thịt)

Kiểm thử tổ hợp (kiểm thử bánh mì kẹp thịt) là việc tổ hợp của các kiểm thử trên xuống và dưới lên. Các kiểm thử trên xuống và dưới lên được tiến hành đồng thời cho tới khi đạt tới làn ranh giới thoả hiệp đã định sẵn.

Hình 5-2-6 Kiểm thử tổ hợp



<Đặc trưng>

- Các mô đun trên đường tổ hợp là chủ đề cho kiểm thử trên xuống trong khi các mô đun dưới đường tổ hợp là chủ đề cho kiểm thử dưới lên.
- Vì các kiểm thử trên xuống và dưới lên có thể được tiến hành đồng thời, nên các kiểm thử có thể được hoàn tất trong thời gian ngắn hơn nhiều.
- Phần khung của chương trình có thể được kiểm thử dễ dàng.
- Xem như hệ thống (chương trình) kiểm thử, thì cả cuống và khiển trình đều được cần tới.

(4) Kiểm thử không tăng dần

Trong kiểm thử này, tất cả các mô đun có việc kiểm thử đơn vị đã được hoàn tất đều được móc nối và cho chạy. Kiểm thử không tăng đại diện là kiểm thử big bang.

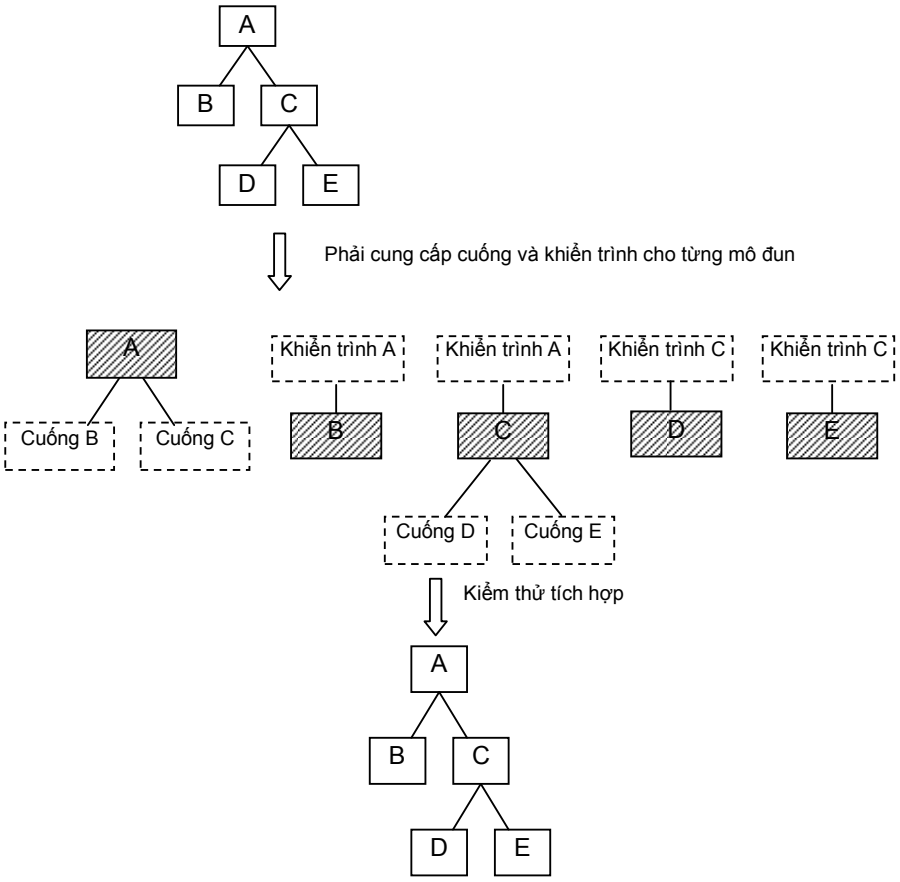
<Đặc trưng>

- Thích hợp cho việc kiểm thử chương trình kích cỡ nhỏ.
- Không cần tới cuống hay khiển trình.
- Nếu lỗi xuất hiện thì khó dò dấu vết chúng ngược về nguyên nhân.

① Kiểm thử Big-bang

Kiểm thử big-bang dùng kĩ thuật của việc thực hiện kiểm thử đơn vị trước hết trên mọi mô đun, rồi móc nối chúng tất cả lại một lúc và thực hiện kiểm thử tích hợp toàn bộ.

Hình 5-2-7 Kiểm thử Big-bang



<Đặc trưng>

- Bởi vì kiểm thử này được tiến hành sau kiểm thử đơn vị nên kết quả có độ tin cậy cao.
- Để thực hiện kiểm thử móc nối thì cuống hay khiển trình là không cần thiết.
- Các mô đun có thể được kiểm thử tất cả ngay một lúc.
- Nếu kiểm thử đơn vị không được hoàn tất thì không thể tiến hành kiểm thử big bang được.
- Khó tìm lỗi trong giao diện giữa các mô đun.
- Sau khi tìm được lỗi, thì việc gỡ lỗi lại lằng nhằng.

5.2.4 Kiểm thử hệ thống

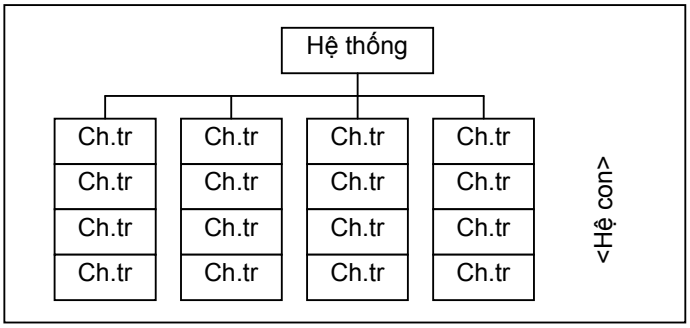
Sau khi các mô đun được móc nối với nhau đã được kiểm thử, thì các kiểm thử được tiến hành theo trình tự của từng chương trình, rồi kiểm thử cho từng hệ con một, và cuối cùng kiểm thử cho toàn bộ hệ thống.
Các kiểm thử được tiến hành sau kiểm thử tích hợp được gọi là kiểm thử hệ thống.

(1) Tổng quan về kiểm thử hệ thống

Kiểm thử hệ thống được tiến hành để kiểm chứng sự phù hợp với thiết kế ngoài. Trong việc

tiến hành kiểm thử này, phần lớn chú ý được dành cho giao diện giữa các hệ con. Kiểm thử hệ thống được gọi là kiểm thử toàn diện, và được tiến hành bởi một nhóm chuyên kiểm thử. Nó là kiểm thử cuối cùng được tiến hành bởi tổ chức phát triển hệ thống.

Hình 5-2-8 Phạm vi của kiểm thử hệ thống



<<Chương trình được tích hợp và toàn bộ hệ thống được kiểm thử>>

(2) Các kiểu kiểm thử hệ thống

Kiểm thử hệ thống được tiến hành để kiểm tra các chức năng và hiệu năng từ nhiều góc độ khác nhau như được nêu dưới đây:

① Kiểm thử tích hợp chương trình/hệ con

Các chương trình được móc nối với nhau, và các hệ con được móc nối với nhau, và các hệ con và giao diện giữa các chương trình được kiểm thử.

② Kiểm thử chức năng

Kiểm thử này được tiến hành để kiểm chứng liệu các yêu cầu chức năng của người dùng có được đáp ứng hay không.

③ Kiểm thử hiệu năng

Kiểm thử này được tiến hành để kiểm chứng thời gian đáp ứng và các khoản mục hiệu năng khác.

④ Kiểm thử vận hành

Giao diện con người (GUI) và các điểm khác liên quan tới vận hành được kiểm tra và kiểm chứng.

⑤ Kiểm thử phục hồi hỏng hóc

Cách thức hệ thống có thể phục hồi từ hỏng hóc và thực hiện lại các chức năng là được kiểm thử.

⑥ Kiểm thử tải

Kiểm thử này được tiến hành để kiểm tra hiệu năng và chức năng của hệ thống khi một khối lượng lớn dữ liệu được đưa vào một lúc, hay khi một tải lớn được áp vào hệ thống.

⑦ Kiểm thử ngoại lệ

Kiểm thử này được tiến hành để kiểm chứng rằng hệ thống có thể giải quyết thích hợp cho các dữ liệu không hợp lệ khi được đưa vào.

⑧ Kiểm thử chịu đựng

Kiểm thử này được tiến hành để kiểm chứng rằng một hệ thống có thể đứng vững nhiều giờ

làm việc liên tục.

5.2.5 Các kiểm thử khác

Cũng còn có kiểm thử vận hành (được tiến hành sau khi kiểm thử hệ thống đã hoàn tất) và kiểm thử rà lại.

(1) Kiểm thử vận hành

Tổ chức của người dùng chịu trách nhiệm tiến hành kiểm thử vận hành. Kiểm thử vận hành là kiểm thử cuối cùng được tiến hành trên hệ thống. Tổ chức của người dùng phải chuẩn bị các trường hợp kiểm thử, tiến hành kiểm thử trong điều kiện vận hành thực tế, và kiểm chứng rằng hệ thống thoả mãn các đặc tả đã được yêu cầu. Bởi vì kiểm thử này dự định để làm cho hệ thống đã xây dựng được tổ chức người dùng chấp nhận, nên nó được gọi là kiểm thử chấp thuận hay kiểm thử chấp nhận.
Bởi vì kiểm thử vận hành được tiến hành bằng cách cho chạy chương trình trên máy đang được dùng cho vận hành nghiệp vụ thực tế, nên phải hết sức chu đáo đừng để làm nhiễu các hoạt động nghiệp vụ.

(2) Kiểm thử rà lại

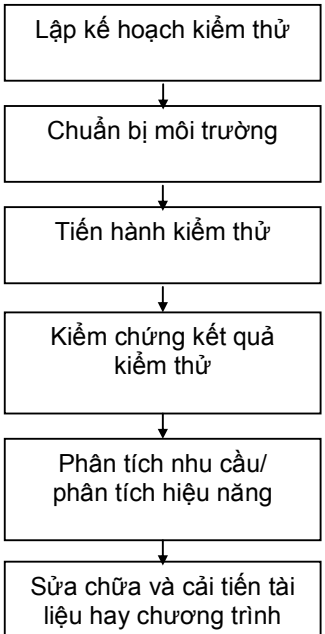
Kiểm thử rà lại có liên quan chặt chẽ với hoạt động bảo trì.
Mục đích của kiểm thử rà lại là kiểm chứng rằng những sửa đổi hệ thống được tiến hành trong công việc bảo trì không ảnh hưởng tới các bộ phận đang hoạt động bình thường của hệ thống.

5.2.6 Kế hoạch và nhiệm vụ kiểm thử

Các kiểm thử đa dạng được tiến hành trong phát triển hệ thống đã được mô tả cả rồi. Mục này giải thích các nhiệm vụ kiểm thử chung và nội dung của công việc được làm để tiến hành kiểm thử.

(1) Nhiệm vụ kiểm thử (tổng quan)

Hình 5-2-9 chỉ ra một tổng quan về các nhiệm vụ kiểm thử.
Hình 5-2-9 Nhiệm vụ kiểm thử

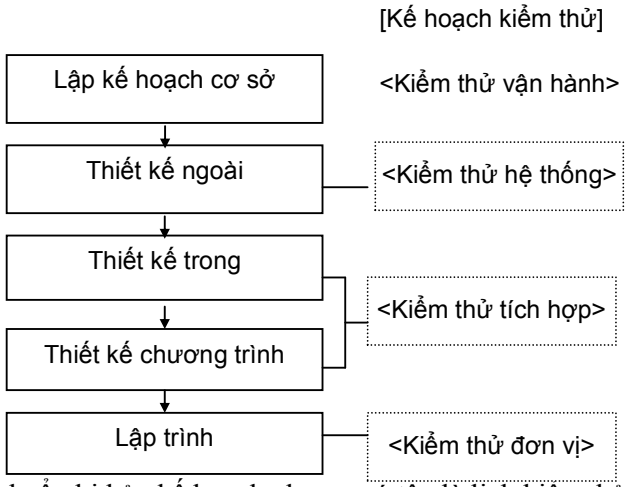


(2) Luồng kiểm thử và nội dung các nhiệm vụ

① Kế hoạch kiểm thử

Cần chuẩn bị bản kế hoạch kiểm thử. Bản kế hoạch kiểm thử có mối quan hệ chặt chẽ với công việc phát triển hệ thống, như được vẽ trong Hình 5-2-10.

Hình 5-2-10 Kế hoạch kiểm thử



Cần chuẩn bị bản kế hoạch chung, có tên là lịch biểu chủ. Các lịch biểu trong pha lập kế hoạch cơ sở cho việc tiến hành kiểm thử được xác định tại điểm này. Mỗi kiểm thử trong Hình 5.2.10 nên được lập lịch với việc đặt thời gian nêu sau đây:

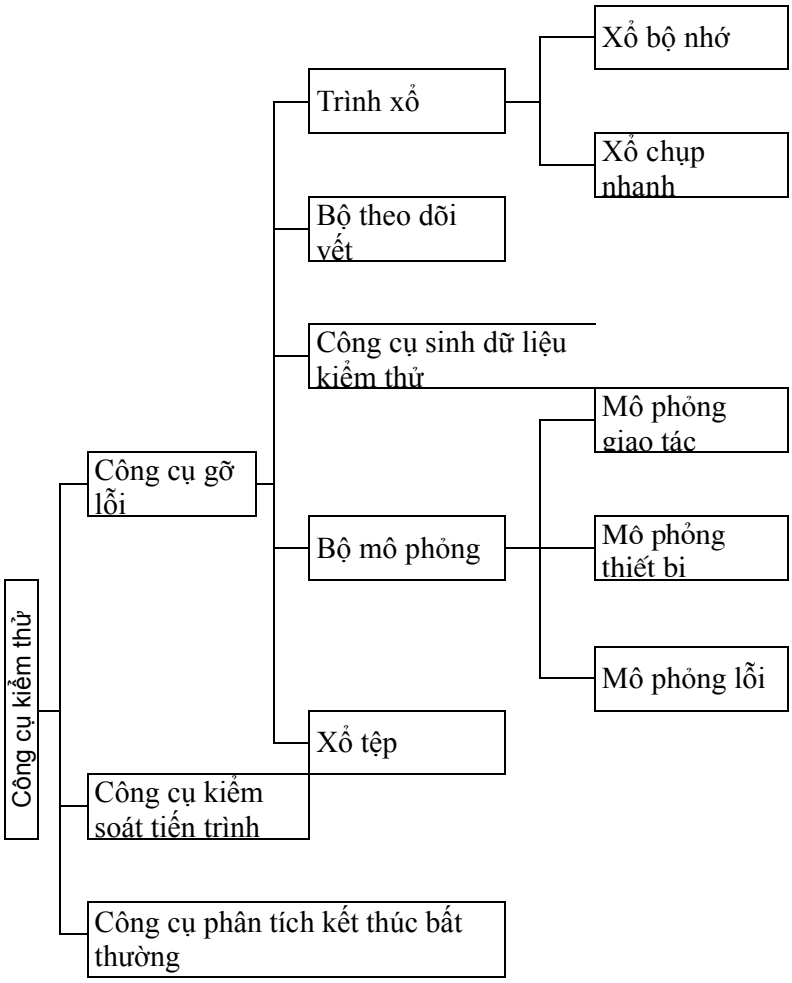
<Khi nào xác định lịch biểu cho từng kiểm thử>

- Kiểm thử vận hành: Được xác định khi kế hoạch cơ bản được tạo ra.
- Kiểm thử hệ thống: Được xác định trong thiết kế trong.
- Kiểm thử tích hợp: Được xác định trong thiết kế trong hoặc thiết kế chương trình.
- Kiểm thử đơn vị (mô đun): Được xác định trong khi lập trình (thiết kế mô đun).

② Chuẩn bị môi trường kiểm thử

Các bước sau đây phải được chuẩn bị trước khi tiến hành từng kiểm thử:

- Hệ thống kiểm thử
- Dữ liệu kiểm thử
- Chương trình kiểm thử (mô đun)
- Công cụ kiểm thử (gói phần mềm)



Hình 5-2-11 Công cụ kiểm thử

a. Thiết kế hệ thống kiểm thử

Có những trường hợp máy được dùng cho các vận hành nghiệp vụ phải được dùng để cho chạy chương trình để kiểm thử. Tuy nhiên, nếu máy này lưu giữ các tệp hay cơ sở dữ liệu vẫn được truy nhập tới trong vận hành hàng ngày, hay nếu một tải lượng lớn bị áp vào máy trong khi kiểm thử, thì việc sản xuất sẽ bị gây rối loạn. Do đó, cần phát triển hay thiết kế cùng hệ điều hành, tệp và cơ sở dữ liệu nhưng sẽ là đối tượng được dùng trong việc chạy sản xuất thực.

b. Thiết kế chương trình kiểm thử và dữ liệu kiểm thử

Phải thiết kế chương trình kiểm thử (cuống và khiển trình). Cũng vậy dữ liệu kiểm thử phải được chuẩn bị và các trường hợp kiểm thử phải được thiết kế bằng việc dùng nhiều phương pháp.

Có thể dùng công cụ phát triển mới có đây để dễ dàng sinh ra dữ liệu kiểm thử dựa trên các tham biến đơn giản. Một công cụ như vậy là rất thuận tiện vì dữ liệu sai có thể được sinh ra một cách có chủ ý.

c. Thiết đặt các công cụ kiểm thử

Các công cụ kiểm thử cần phải được thiết đặt. Các chương trình tiện ích của hệ điều hành nói chung được dùng làm công cụ kiểm thử. Các gói phần mềm được thiết kế làm công cụ kiểm thử bây giờ rất sẵn có. Môi trường vận hành của gói như vậy nên được kiểm tra và đưa vào để làm giảm chi phí phát triển, và làm tăng tính hiệu quả và chất lượng của kiểm thử.
Hình 5-2-11 đưa ra một số công cụ kiểm thử.

1) Công cụ gỡ lỗi

- Trình xổ ra
  Trình xổ ra đưa ra nội dung của bộ nhớ hay thanh ghi.
  - Xổ bộ nhớ: Viết ra nội dung của bộ nhớ hay thanh ghi khi việc kết thúc bất thường xuất hiện.
  - Xổ chụp nhanh: Với một lệnh gỡ rối nhúng trong chương trình, và nội dung của bộ nhớ hay thanh ghi được viết ra mỗi lần lệnh này được thực hiện
- Bộ dò vết
  Bộ dò vết cũng còn được gọi là chương trình dò vết. Mỗi lần một lệnh được thực hiện và điều kiện nào đó được thoả mãn, thì bộ dò vết được kích hoạt và ghi lại nội dung của thanh ghi và địa chỉ của vùng bộ nhớ được tham chiếu.
- Công cụ sinh dữ liệu kiểm thử
  Công cụ sinh dữ liệu kiểm thử tự động sinh ra dữ liệu kiểm thử dựa trên các tham biến do người dùng cung cấp.
- Bộ mô phỏng
  - Bộ mô phỏng giao tác mô phỏng cho hệ thống xử lí giao tác.
  - Bộ mô phỏng thiết bị mô phỏng cho một thiết bị cuối.
  - Bộ mô phỏng hỏng hóc mô phỏng cho các hoàn cảnh hỏng hóc.
- Xổ tệp
  Chương trình xổ tệp ghi ra nội dung của một tệp được ghi trên thiết bị nhớ như đĩa từ hay băng từ.

2) Công cụ kiểm soát tiến trình kiểm thử
Công cụ kiểm soát tiến trình kiểm thử có thể hỗ trợ cho mọi tiến trình kiểm thử từ lập kế hoạch kiểm thử và thiết kế cho tới gỡ lỗi.

3) Công cụ phân tích kết thúc bất thường
Công cụ phân tích kết thúc bất thường có thể tìm ra nguyên nhân của sự kết thúc bất thường xuất hiện cho chương trình, và nêu ra những hành động sửa đổi.

Các công cụ kiểm thử trên nên được sử dụng nhiều nhất có thể được có tính tới ngân sách có sẵn. Chúng là những công cụ hiệu quả có thể đóng góp cải tiến năng suất. Kết quả là, chất lượng của hệ thống sẽ tăng lên, chi phí bảo trì sau khi hệ thống trở nên vận hành có thể được giảm bớt, và có thể trông đợi đầu tư quay vòng tốt hơn.

Cũng vậy, các điểm sau nên được xem xét trước khi tiến hành kiểm thử:

- Tạo ra một chức năng cho việc xử lí dữ liệu kiểm thử trong toàn bộ bằng phương tiện JCL (job control language), các lệnh lớp vỏ hay tệp xử lí theo lô.
- Cải thiện hỏng hóc, mạng và các môi trường mô phỏng khác (bằng việc dùng công cụ kiểm thử)
- Tự động hoá xử lí tương tác

③ Tiến hành kiểm thử

Nhiều loại kiểm thử phải được tiến hành trong điều kiện môi trường thực hiện được chuẩn bị tốt.

④ Kiểm tra kết quả của kiểm thử

Kiểm thử được tiến hành tương ứng với các kế hoạch kiểm thử và đặc tả kiểm thử, và kết quả của việc kiểm thử phải được kiểm tra lại.

⑤ Phân tích hỏng hóc, phân tích hiệu năng

Các lỗi và hỏng hóc bị phát hiện phải được phân tích chặt chẽ bằng việc dùng các công cụ và các kĩ thuật kiểm tra chất lượng đa dạng.

⑥ Sửa chữa và cải tiến tài liệu và chương trình gốc

Nếu lỗi hay sai sót thiết kế được tìm thấy và nếu chúng có thể được sửa chữa ngay lập tức, thì chương trình nguồn phải được sửa chữa hay cải tiến. Phần của tài liệu thiết kế liên quan tới các lỗi hay sai sót thiết kế như vậy cũng phải được sửa chữa. Nếu chỉ chương trình nguồn được sửa chữa, thì sự nhất quán giữa chương trình nguồn và tài liệu thiết kế sẽ bị mất, và rắc rối có thể xuất hiện khi công việc kiểm chứng các phần khác hay việc bảo trì được tiến hành.

⑦ Lấy hành động thích hợp sau một kiểm thử hoàn tất

Sau khi một kiểm thử đã được hoàn tất và chương trình đã được sửa chữa, thì các hành động sau phải được tiến hành:

a. Quản lí tiến độ kiểm thử và báo cáo

   Người chịu trách nhiệm phải báo cáo về tiến độ của việc kiểm thử bằng việc dùng báo cáo công việc hàng tuần hay báo cáo kiểm thử. Sau khi kiểm thử được hoàn tất, người đó phải báo cáo kết quả của việc kiểm thử bằng việc dùng báo cáo hoàn thành kiểm thử.

b. Kiểm soát dữ liệu liên quan tới hỏng hóc

   Dữ liệu về lỗi hay khiếm khuyết được tìm ra trong khi kiểm thử phải được tích luỹ lại. Cũng vậy, lỗi xuất hiện trong tình huống nào, hành động sửa chữa nào đã tiến hành và các thông tin chi tiết khác phải được cất giữ.

c. Xem lại tài liệu vận hành thượng lưu

   Một sai lầm được tìm ra trong giao diện con người hay trong các giao diện giữa các hệ con có thể qui cho thiết kế trong, như được mô tả trong mục ⑥ trên. Không chỉ phần của chương trình dẫn tới sai lầm đó phải được sửa lại, mà cả bản thân tài liệu thiết kế trong cũng phải được chữa lại để ngăn cản cùng sai lầm đó xảy ra nữa.
   Để ngăn cản việc lặp lại các lỗi gây ra bởi một sai lầm phạm phải trong tiến trình ngược dòng trước, một chương trình, hay một phần của tài liệu thiết kế liên quan tới sai lầm đó cũng như tài liệu vận hành tương ứng, phải được sửa lại hay xem xét lại để ngăn cản sự xuất hiện nữa của cùng sai lầm.

Bài tập

Q1 **Trong khi tiến hành các kiểm thử trong giai đoạn phát triển hệ thống, các kiểm thử được tiến hành theo thứ tự từ các đơn vị nhỏ tới đơn vị lớn, và kết quả của việc kiểm thử được tích luỹ lại như dữ liệu. Trật tự các kiểm thử nào sau đây là thích hợp nhất?**

a. Kiểm thử hệ thống → kiểm thử tích hợp → kiểm thử đơn vị
b. Kiểm thử hệ thống → kiểm thử đơn vị → kiểm thử tích hợp
c. Kiểm thử đơn vị → kiểm thử tích hợp → kiểm thử hệ thống
d. Kiểm thử đơn vị → kiểm thử hệ thống → kiểm thử tích hợp

Q2 **Thuật ngữ nào là thích hợp nhất cho việc kiểm thử được tiến hành với sự chú ý nhất được dành cho cấu trúc bên trong của chương trình và thuật toán?**

a. Kiểm thử hệ thống
b. Kiểm thử trên xuống
c. Kiểm thử hộp đen
d. Kiểm thử hộp trắng
e. Kiểm thử dưới lên

Q3 **Kĩ thuật nào sau đây là để chuẩn bị dữ liệu kiểm thử, và kiểm thử các chức năng của chương trình với sự chú ý nhiều nhất được dành cho mối quan hệ giữa dữ liệu vào và kết quả đưa ra?**

a. Kiểm thử trên xuống
b. Kiểm thử hộp đen
c. Kiểm thử dưới lên
d. Kiểm thử hộp trắng

Q4 **Mô tả nào sau đây là thích hợp cho kiểm thử tích hợp được tiến hành trong tiến trình phát triển hệ thống, ngay sau kiểm thử đơn vị (kiểm thử mô đun) được hoàn tất?**

a. Kiểm chứng rằng hệ thống có thể thực hiện không có lỗi nào cho tất cả các chức năng được xác định trong tài liệu thiết kế ngoài
b. Kiểm chứng rằng tập các mục tiêu về thời gian xử lí và mục tiêu thời gian đáp ứng đã được đạt tới
c. Kiểm chứng rằng không có vấn đề gì trong các kiểu và số thiết bị vào và ra và thiết bị truyền thông được ghép nối
d. Kiểm chứng rằng không có vấn đề gì với giao diện giữa các mô đun, là các cấu phần của chương trình
e. Kiểm chứng rằng việc cho chạy nhiều việc và ghép nối đồng thời các thiết bị cuối có thể được thực hiện như đã được xác định

Q5 **Giải thích nào là đúng về kiểm thứ dưới lên, một trong những kĩ thuật kiểm thử?**

a. Kiểm thử được tiến hành bằng cách móc nối các mô đun theo trật tự mô đun thấp tới cao. Các khiển trình được cần tới làm cái thay thế cho các mô đun mức cao chưa hoàn tất.

b. Từng mô đun riêng lẻ được kiểm thử. Khi tất cả các mô đun đều đã được kiểm thử, thì chúng được móc nối và kiểm thử.
c. Kiểm thử được tiến hành bằng việc móc nối các mô đun theo thứ tự từ mô đun cao xuống mô đun thấp. Cuống được cần tới như cái thay thế cho các mô đun cấp thấp chưa hoàn tất.
d. Các kiểm thử được tiến hành theo trật tự kiểm thử đơn vị, tích hợp, hệ thống và vận hành.

Q6 Mô tả nào là đúng cho dữ liệu kiểm thử được dùng để giám định chương trình?

a. Các trường hợp kiểm thử được chuẩn bị trước, và dữ liệu kiểm thử có thể đáp ứng các yêu cầu được xác định trong trường hợp kiểm thử được chuẩn bị.
b. Chỉ dữ liệu kiểm thử có thể được xử lí đúng mới được chuẩn bị như các tiến trình kiểm thử.
c. Như dữ liệu được dùng cho kiểm thử, quãng 20% khối lượng dữ liệu cần xử lí trong các thao tác thực tế mới được chuẩn bị.
d. Dữ liệu kiểm thử bị bác bỏ như lỗi trong giai đoạn đưa vào không cần phải được cung cấp.

Q7 Kĩ thuật gỡ lỗi nào để ghi ra nội dung của các biến hay thanh ghi mỗi lần một câu lệnh đặc biệt được thực hiện?

a. Walk-through
b. Ảnh chụp nhanh
c. Bộ sinh dữ liệu kiểm thử
d. Khiển trình

6 Cập nhật vận hành và phát triển hệ thống

Mục đích của chương

Hiểu thiết kế chương trình bằng cách sử dụng tiếp cận hướng đối tượng.

① Hiểu định nghĩa lớp và các mối quan hệ
② Hiểu cách ánh xạ thiết kế vào hình thức thực hiện
③ Hiểu thiết kế các dịch vụ Web bằng cách sử dụng tiếp cận hướng đối tượng

Giới thiệu

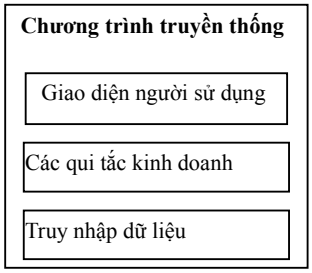
Rất nhiều gói phần mềm được phát triển đi cùng với hình thức đối tượng nào đó. Khả năng tái sử dụng các lớp mà không cần lo lắng về thân chúng cho phép các hệ thống kết hợp lỏng lẻo được phát triển một cách dễ dàng. Trong chương này, chúng ta sẽ giải thích về thiết kế chương trình sử dụng phương pháp hướng đối tượng.

6.1 Thiết kế chương trình

6.1.1 Thiết kế chương trình hướng đối tượng

(1) Tổ chức của chương trình hướng đối tượng

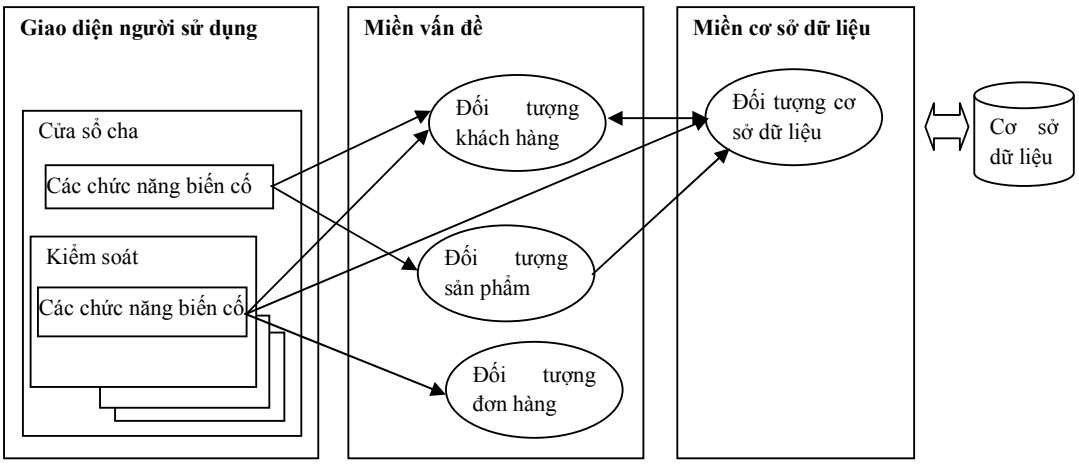
Truy nhập vào giao diện người sử dụng, đọc cơ sở dữ liệu và các qui tắc kinh doanh, tất cả đều được tổ hợp với nhau trong chương trình truyền thống.



Khi cách tiếp cận hướng đối tượng được chấp nhận cho thiết kế chương trình, các cấu phần riêng lẻ phải được tách bạch. Hầu hết các giao diện người sử dụng sử dụng mô hình điều khiển theo biến cố. Các hành động thực hiện trong thế giới thực được dịch thành lời gọi chức năng. Bộ giải quyết biến cố thích hợp trong giao diện người sử dụng được thay đổi để xử lí hành động. Điều này dẫn tới các kiểu lớp sau:

① Giao diện người sử dụng
② Các lớp miền vấn đề
③ Các lớp miền cơ sở dữ liệu

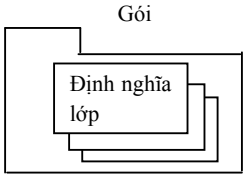
Bằng cách chia thiết kế chương trình thành 3 miền này, điều đó cho phép hiểu và tạo các miền dễ dàng hơn.



Điều này cho phép việc nối lỏng lẻo giữa các phần khác nhau của chương trình bằng việc chia các định nghĩa lớp vào ba miền này. Giao diện người dùng ở bên máy khách để cho phép người dùng tương tác với hệ thống. Các lớp miền vấn đề được dùng để bao bọc các qui tắc kinh doanh. Lớp cơ sở dữ liệu được dùng để cô lập vị trí của cơ sở dữ liệu làm cho nó thành trong suốt với hệ thống. Vị trí của các lớp miền vấn đề có thể ở trên máy phục vụ hay máy khách. Vì từng máy khách dùng một tập các giá trị khác nhau, nên trong thực hành điều thông thường là cấp cho các đối tượng miền vấn đề vào máy khách.

(2) Lớp và đối tượng

Đối tượng là thể nghiệm của lớp. Bạn có thể coi lớp như định nghĩa. Các lớp được nhóm với nhau như một gói.



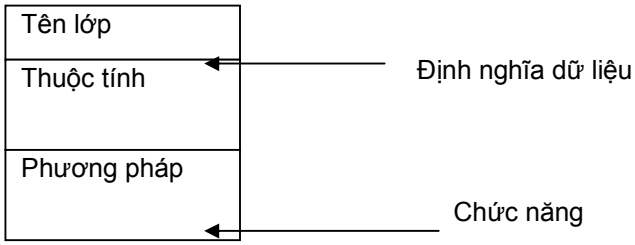
Lớp gồm 2 phần

① Thuộc tính

Thuộc tính chứa định nghĩa dữ liệu cho lớp.

② Phương pháp (Methods)

Phương pháp trở thành các chức năng trong lớp.

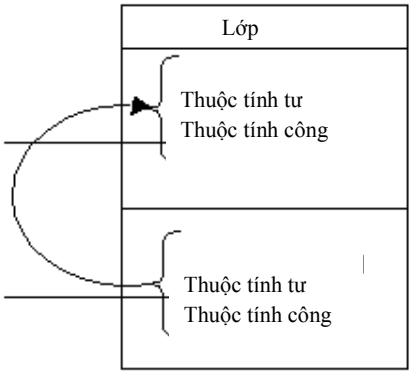


Vùng truy nhập

Có 3 kiểu vùng truy nhập.

a. Công (public)

Truy nhập công nghĩa là các phương pháp và các thuộc tính có thể được truy nhập tới từ bên ngoài.

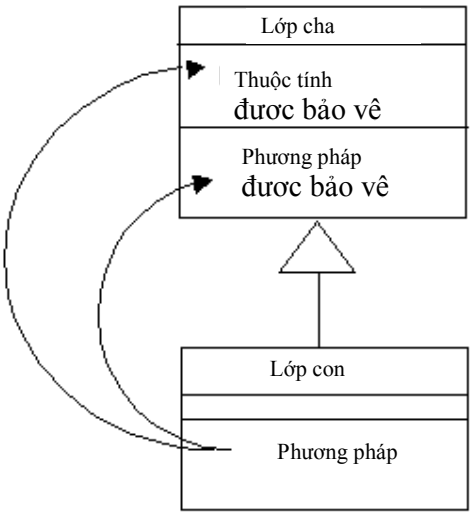


b. Truy nhập tư (Private)

Điều này nghĩa là các phương pháp hay các thuộc tính chỉ có thể được truy nhập bởi các phương pháp nằm bên trong đối tượng. Điều này nghĩa là ngay cả các đối tượng của cùng một lớp cũng không thể truy nhập được tới các phương pháp tư của các đối tượng khác. Vì dữ liệu trong mỗi lớp được quản lý bởi bản thân đối tượng này, nên điều này có nghĩa là các thuộc tính thường là tư và các phương pháp là công

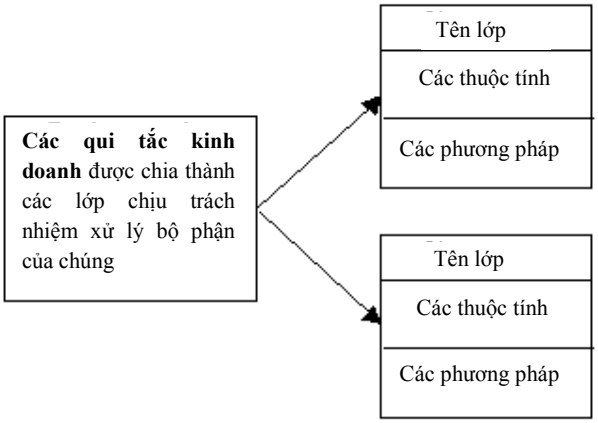
c. Được bảo vệ (Protected)

Điều này chỉ áp dụng nếu kế thừa được cài đặt. Truy nhập được bảo vệ nghĩa là các con cháu của lớp này có thể truy nhập các phương pháp và các thuộc tính đó.



④ Các lớp miền vấn đề

Các lớp miền vấn đề biểu diễn cho động cơ kinh doanh. Các qui tắc kinh doanh được được bao bọc trong các lớp miền vấn đề

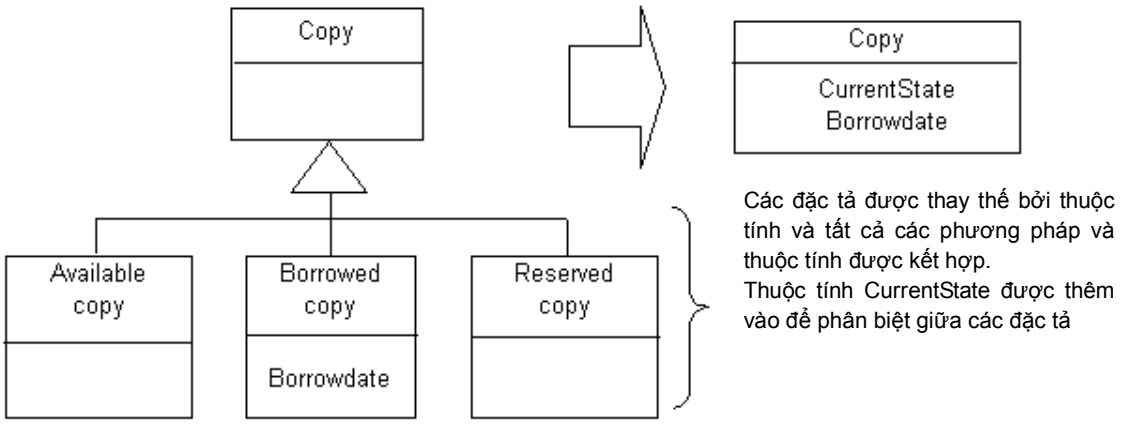


(3) Các thuộc tính

Các thuộc tính biểu diễn dữ liệu mà mô tả cho lớp miền vấn đề. Các kiểu thuộc tính khác là
Thuộc tính trạng thái
Thuộc tính quan hệ với khóa ngoại
Bản số của đối tượng

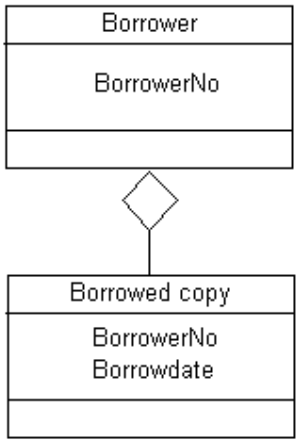
① Thuộc tính trạng thái

Các lớp đặc biệt hoá biểu diễn các trạng thái có thể được định nghĩa bằng cách sử dụng thuộc tính trạng thái.



② Thuộc tính quan hệ khóa ngoại

Nếu một lớp có một liên kết hoặc một kết tập với một lớp khác, thì khóa ngoại xuất hiện trong lớp thiết kế. Ví dụ

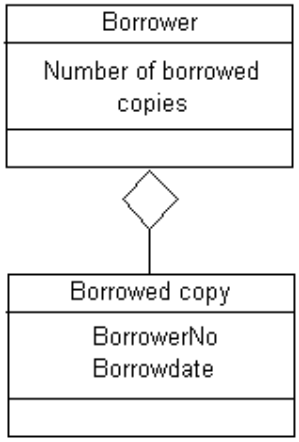


Nếu BorrowerNo là danh xưng đối tượng cho borrower, thì nó cũng xuất hiện như một thuộc tính trong lớp Borrowed copy.

③ Bản số của đối tượng (Cardinality of the object)

Các thuộc tính kiểu số là ứng viên tốt cho việc xác định các trạng thái giới hạn. Bản số của đối tượng được biểu diễn như một thuộc tính trong lớp.

Ví dụ về người mượn Borrower và tài liệu được mượn Borrowed copy



Một thuộc tính diễn tả số các bản sao được mượn (Number of borrowed copies) có thể được thêm vào lớp

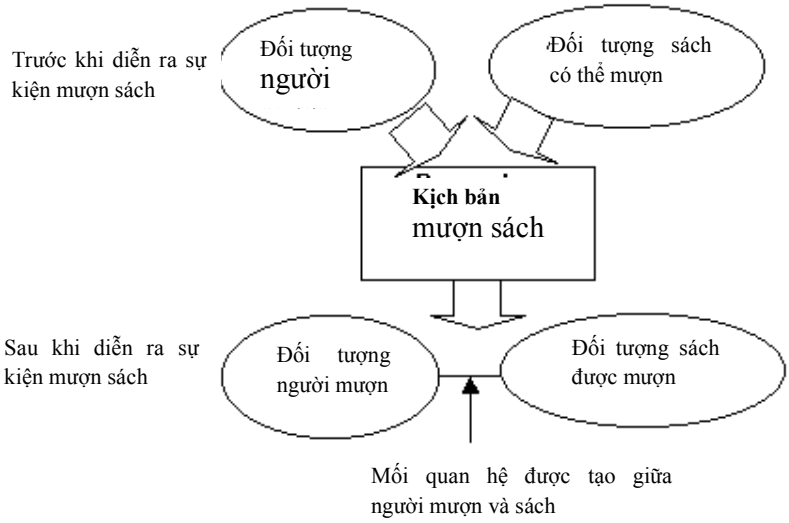
borrower . Tổng số các bản sao được mượn (number of borrowed copies) là hữu hạn. Thuộc tính này có thể được sử dụng để xác định xem người mượn có vượt quá giới hạn cho phép hay không.

(4) Các kiểu phương pháp trong các lớp miền vấn đề

Các phương pháp được tìm thấy trong các lớp miền vấn đề có thể được phân loại là
Các phương pháp kiểm chứng trạng thái
Các phương pháp thay đổi trạng thái
Các phương pháp trao đổi dữ liệu

① Các phương pháp kiểm chứng trạng thái

Xét kịch bản mượn sách trong hệ thống thư viện. Người mượn và tài liệu phải thoả mãn một số điều kiện trước khi được phép tham gia vào trong kịch bản này. Sau kịch bản này, có sự thay đổi trạng thái của các đối tượng như mối quan hệ được tạo ra giữa người mượn và tài liệu.



Phương pháp kiểm chứng trạng thái được sử dụng để xác định trạng thái ban đầu của đối tượng. Các đối tượng tài liệu với trạng thái có sẵn được phép tham gia trong việc mượn sách. Điều này nghĩa là phương pháp kiểu như IsAvailable cho cái ra kiểu True/False có thể được sử dụng để xác định xem đối tượng tài liệu là trong trạng thái sẵn có hay không.
Ví dụ về phương pháp kiểm chứng trạng thái trong lớp Copy (ví dụ VB)
public Sub IsAvailable(OutReply As Boolean)

② Phương pháp thay đổi trạng thái

Phương pháp này được sử dụng để làm cho đối tượng thay đổi trạng thái của nó.
Ví dụ trong trường hợp kịch bản mượn sách
Đối tượng tài liệu được thay đổi từ trạng thái sẵn có sang trạng thái đã được mượn. Điều này nghĩa là khả năng nhận diện đối tượng người mượn liên kết phải được truyền qua đối tượng tài liệu.
Ví dụ về phương pháp thay đổi trạng thái trong lớp Copy
public Sub Borrow(InBorrower As Borrower)

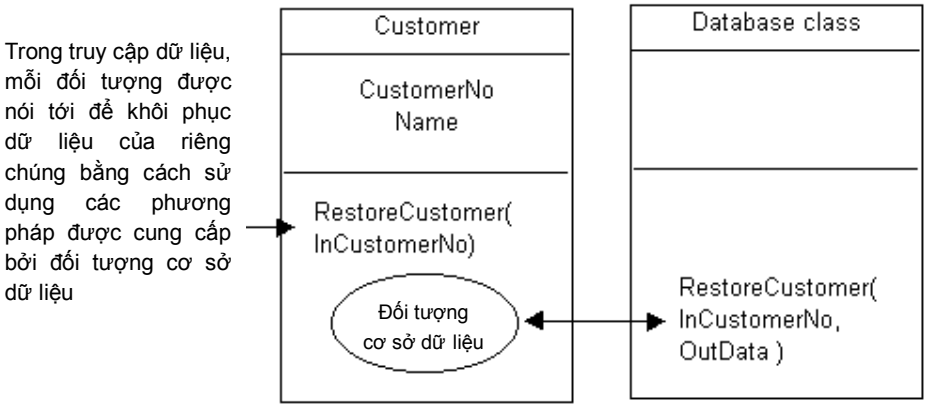
③ Phương pháp trao đổi dữ liệu

Phương pháp này biểu diễn sự trao đổi thông tin giữa các đối tượng. Giao diện người sử dụng có thể yêu cầu các giá trị được nêu ra cho thế giới thực. Thay vì tạo nhiều phương pháp trao đổi dữ liệu, một cấu trúc trao đổi có thể được định nghĩa như một tham biến. Hai phương pháp có thể được định nghĩa để biểu diễn dữ liệu truyền đi hoặc dữ liệu trả lại.

(5) Sự phụ thuộc giữa các đối tượng

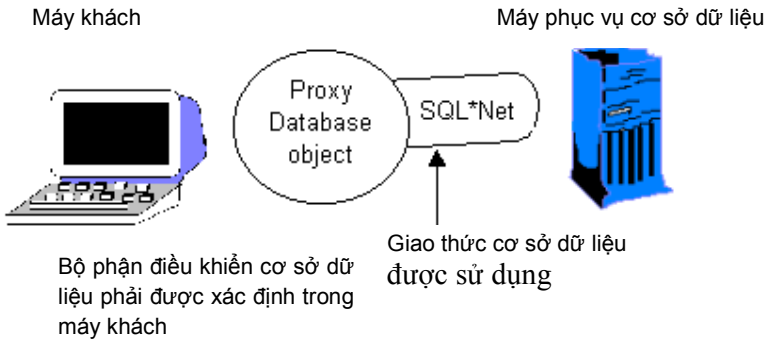
Mỗi lớp có vùng trách nhiệm riêng của nó. Khi một khách hàng cụ thể được tích trữ, tiêu chuẩn truy nhập như giá trị khóa được truyền vào một phương pháp trong lớp khách hàng (Customer). Lớp khách hàng sẽ

không đọc trực tiếp cơ sở dữ liệu. Việc viết và đọc cơ sở dữ liệu thuộc trách nhiệm của lớp Database. Thay vì thế một đối tượng của lớp database được tạo trong thân của phương pháp Customer. Phương pháp được cung cấp bởi đối tượng database rồi được thực hiện bằng việc truyền tiêu chuẩn truy nhập được yêu cầu và nhận lại dữ liệu. Không giống như các chương trình truyền thống, dữ liệu được sử dụng để thiết lập các giá trị trong đối tượng miền vấn đề, tức là customer.

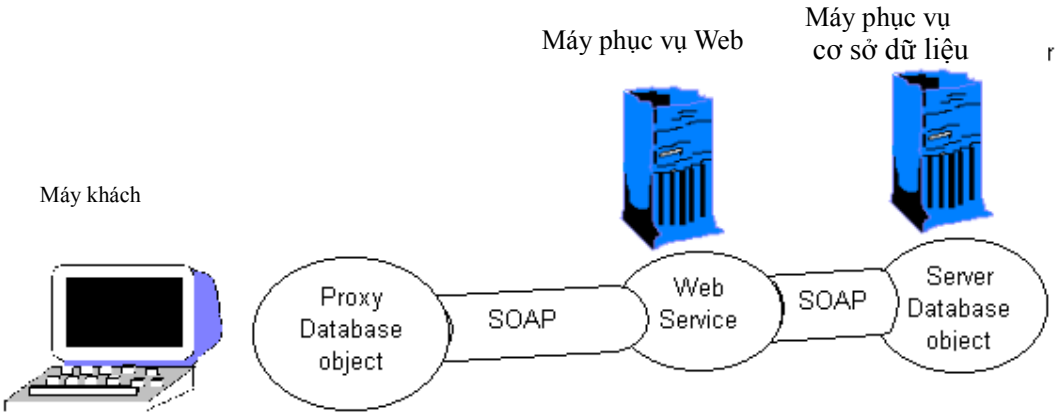


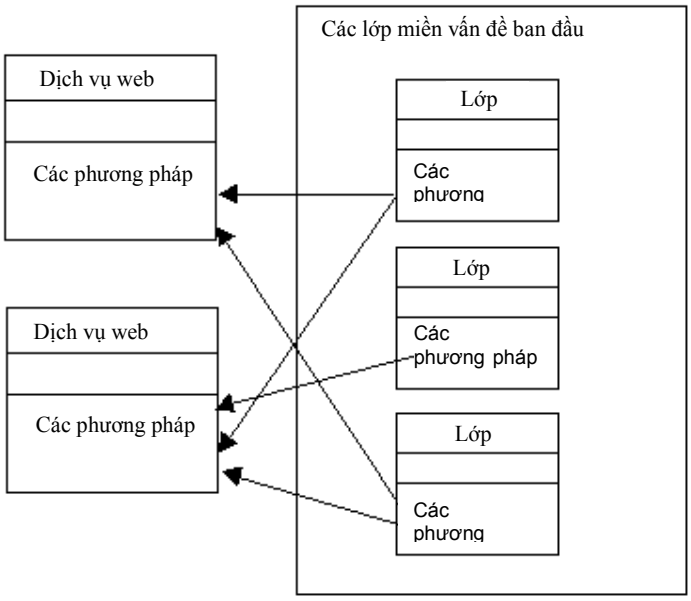
(6) Lớp cơ sở dữ liệu (Database)

Lớp cơ sở dữ liệu được sử dụng để che giấu vị trí của máy phục vụ cơ sở dữ liệu. Nó được xử lí giống như máy uỷ quyền proxy cho máy phục vụ. Trong môi trường khách/phục vụ, lớp cơ sở dữ liệu sẽ thực tế giao tiếp với máy phục vụ cơ sở dữ liệu. Nhược điểm là giao diện máy khách cơ sở dữ liệu phải được cài đặt.



Ngày nay, các kiến trúc đa bên đã thành thông dụng hơn. Trong kiến trúc đa bên, đối tượng cơ sở dữ liệu trong máy khách thực tế là một uỷ quyền proxy sử dụng giao thức chuẩn như SOAP để truy nhập các dịch vụ web trên máy phục vụ. Sau đó, các dịch vụ web có thể truy nhập máy phục vụ cơ sở dữ liệu hoặc máy phục vụ web khác để đi tới máy phục vụ cơ sở dữ liệu. Các dịch vụ web có thể được coi như một tổ hợp các phương pháp lớp miền vấn đề được yêu cầu mà đã được đưa ra để sử dụng.




Các lớp miền vấn đề ban đầu
Dịch vụ web
Các phương pháp
Dịch vụ web
Các phương pháp
Lớp
Các phương
Lớp
Các phương pháp
Lớp
Các phương

Trả lời bài tập

Trả lời cho Quyển 3 Chương 1 (Cấu trúc dữ liệu)

Danh sách đáp án

Đáp án

Q 1:	c	Q 2:	b	Q 3:	b	Q 4:	d	Q 5:	c
Q 6:	c	Q 7:	d	Q 8:	b	Q 9:	a	Q 10:	c
Q 11:	b	Q 12:	c						

Trả lời và mô tả

Q1

Trả lời

c. 190

Mô tả

Vị trí của a[5,6] trong mảng như sau.

a[1,1]	a[1,2]		…				…		a[1,10]
a[2,1]	a[2,2]		…				…		a[2,10]
a[3,1]	a[3,2]		…				…		a[3,10]
a[4,1]	a[4,2]		…				…		a[4,10]
a[5,1]	a[5,2]	a[5,3]	a[5,4]	a[5,5]	a[5,6]				

Nó ở vị trí thứ 46.

Địa chỉ một hai của (a[1,2]) là 100 + 2*1=102

Và địa chỉ một ba của (a[1,3]) là 100 + 2*2=104.

Vì vậy địa chỉ một 46 là

100 + 2*45=190

Q2

Trả lời

b. Con trỏ cho Shizuoka được đặt là 70 và con trỏ cho Atami được đặt là 150

Mô tả

Danh sách một chiều có thể được hình dung như sau:

10 Tokyo-->50 ShinYokohama-->90 Atami-->70 Hamamatsu-->30 Nagoya

Sau khi chèn Shizuoka giữa Atami và Hamamatsu, danh sách một chiều sẽ như sau

10 Tokyo-->50 ShinYokohama-->90 Atami-->150 Shizuoka-->70 Hamamatsu-->30 Nagoya

Vì vậy, con trỏ của Shizuoka sẽ là 70. Con trỏ của Atami sẽ là 150.

Con trỏ đầu
10

Địa chỉ	Dữ liệu	Con trỏ
10	Tokyo	50
30	Nagoya	0
50	Shin Yokohama	90
70	Hamamatsu	30
90	Atami	150
150	Shizuoka	70

Q3

Trả lời

b. Hàng đợi

Mô tả

Hàng đợi được gọi là hệ thống vào trước ra trước (FIFO) → Đây là câu trả lời

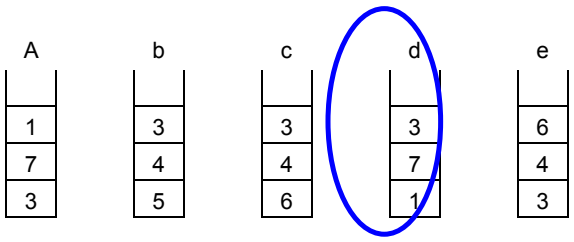
Ngăn xếp được cho là hệ thống vào sau ra trước (LIFO).

Cây nhị phân là loại cấu trúc cây mà thứ tự chèn và thứ tự truy cập là độc lập (không giống FIFO hoặc LIFO)

Đống là loại cây nhị phân.

Q4

Trả lời



Mô tả

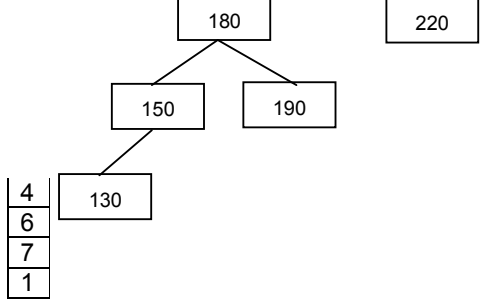
Ngăn xếp quảnl ý dữ liệu theo FILO(vào trước ra sau).

PUSH 1-->PUSH 5-->POP-->PUSH 7-->PUSH 6-->PUSH 4-->POP-->POP-->PUSH 3

PUSH 1-->PUSH 5

POP

PUSH 7-->PUSH 6-->PUSH 4



4
6
7
1

POP-->POP

7
1

PUSH 3

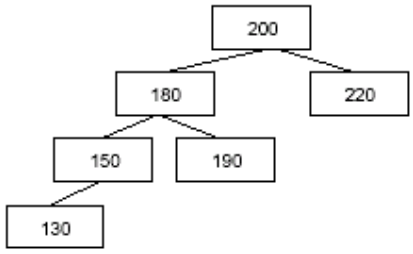
3
7
1

Q5

Trả lời

c. 4

Mô tả



Hình 1 Cây nhị phân

Chỉ số	Giá trị	Con trỏ 1	Con trỏ 2
1	200	3	2
2	220	0	0
3	180	5	a
4	190	0	0
5	150	6	0
6	130	0	0

Hình 2
Mảng diễn tả cây nhị phân

Trong hình 1 ở trên, mỗi nút diễn tả bằng một số chỉ số, con trỏ 1 xác định giá trị chỉ số của nó từ nút con bên trái, con trỏ 2 xác định giá trị chỉ số của nó từ nút con bên phải.
"a" nghĩa là giá trị trị số của nút bên phải của 180 tức là 190 và giá trị chỉ số là 4.

Q6

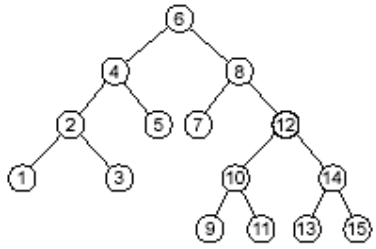
Trả lời

c. 13

Mô tả

Trong cây nhị phân, mỗi nút phải thỏa mãn điều kiện sau.
Giá trị khóa lớn nhất của các nút bên trái nút < Giá trị khóa của nút < Giá trị khóa nhỏ nhất của các nút bên phải nút
Vì vậy

Nút có giá trị 12 phải có giá trị lớn hơn 10 và nhỏ hơn 14.



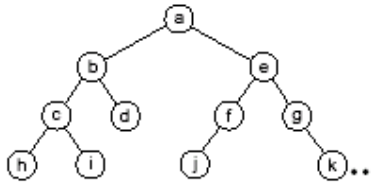
Q7

Trả lời

d. hicdbjfkgea

Mô tả

Tìm kiếm là thực hiện sử dụng phương pháp thứ tự ví trị, nghĩa là
Điểm bắt đầu ở dưới cùng bên trái, phía phải mỗi nút được theo dõi một cách tuần tự.
Vì vậy, h, i và c được đưa ra trước.

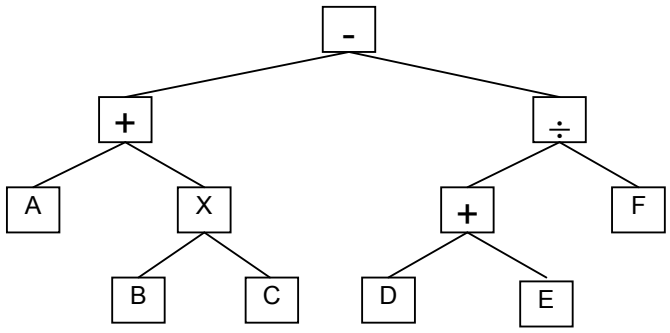


Q8

Trả lời

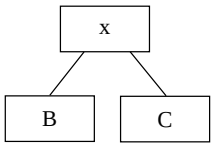
b. $A + B \times C - (D + E) \div F$

Mô tả

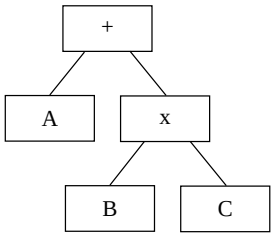


Ký hiệu

Vì vậy,



diễn tả B x C,



diễn tả A + B x C

Bằng cách giải thích dựa theo hướng dẫn như trên, câu trả lời là

A + B x C - (D + E) / F

Q9

Trả lời

a. Chia ra và gộp lại các nút để cho phép chiều sâu phân cấp trở thành như nhau.

Mô tả

a là đúng.

b phản ánh băm

(b. Nhận diện vị trí nơi dữ liệu được lưu giữ bằng việc dùng một hàm nào đó và giá trị khoá.)

c mô tả các tệp truy cập tuần tự

(c. Chỉ có thể truy nhập tuần tự tới dữ liệu đầu và dữ liệu tiếp đó.)

d diễn tả các tệp tổ chức phân hoạch

(d. Có danh mục và một thành viên. Thành viên là tệp được tổ chức tuần tự.)

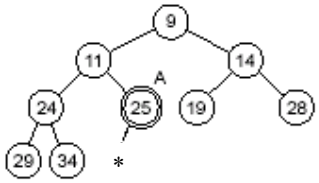
Q10

Trả lời

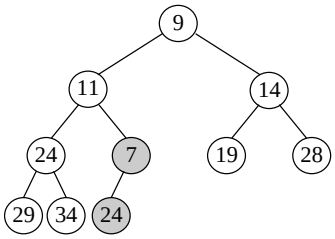
c. 11

Mô tả

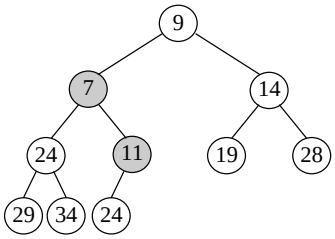
Trong câu hỏi này cần tìm vị trí A sau khi 7 được chèn vào vị trí “*”.



1) Đổi chỗ 7 và 25 vì 7 < 25

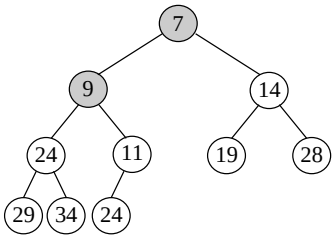


2) Đổi chỗ 7 và 11 vì 7 < 11



3) Đổi chỗ 7 và 9 vì 7 < 9.

Điều này htoải mãn yêu cầu hoàn thành việc đổi chỗ.



Sau khi đổi chỗ như trên, phần tử trong vị trí A là 11. Vì vậy, câu trả lời là c.

Q11

Trả lời

b. 2

Mô tả

Chức năng băm

mod(a1＋a2＋a3＋a4＋a5, 13)

mod(54321,13)=mod(5+4+3+2+1,13)=2

Vì vậy chỉ số của 54321 là 2.

Q12

Trả lời

c. 0.7

Mô tả

Bảng băm với 10 phân tử

Vì vậy xác xuất từ đồng nghĩa không xảy ra cho các giá trị 5 là

9/10 x 8/10 x 7/10 x 6/10 = 3024/10000

Xác xuất xung đột xảy ra là

1-0.3024=0.6976

Trả lời cho Quyển 3 Chương 2 (Các thuật toán)

Danh sách đáp án

Đáp án

Q 1:	a	Q 2:	d	Q 3:	c	Q 4:	b	Q 5:	d
Q 6:	d	Q 7:	e	Q 8:	e	Q 9:	d	Q 10:	c
Q 11:	a	Q 12:	b						

Trả lời và mô tả

Q1

Trả lời

a. Tuyến tính

Mô tả

a. Tuyến tính

Thuật toán tìm kiếm tuyến tính tìm các phần tử tuần tự từ đầu tới cuối bảng → Đây là câu trả lời

b. Nhị phân

Phương pháp tìm kiếm nhị phân là phương pháp thu hẹp dữ liệu đích xuống trong khi chia vùng tìm kiếm thành hai phần. Các phần tử không được tìm kiếm một cách tuần tự.

c. Băm

Băm là cách sử dụng cấu trúc dữ liệu kiểm mảng. Sử dụng băm, bạn có thể truy cập trực tiếp tới dữ liệu cụ thể sử dụng khóa không truy cập dữ liệu bản ghi đối xứng một một.

d. Đống

Đống là loại cây tìm kiếm nhị phân. Nó là cây nhị phân hoàn hảo, có mối liên hệ về kích cỡ giữa nút cha và nút con. Đống khác với tìm kiếm nhị phân ở chỗ đống không có mối quan hệ về kích cỡ giữa các nút anh em.

Q2

Trả lời

d. $X = a_i$

Mô tả

Chỉ số	1	2	3	...	i	...	n	n+1
Giá trị	a_1	a_2	a_3	...	a_i	...	a_n	X

Lặp lại từ bước 2 tới bước 4.

Khoảng trống ☐ diễn tả điều kiện lặp.

Điều kiện tồn tại là “giá trị được tìm thấy” hoặc “ vị trí cuối được tìm”.

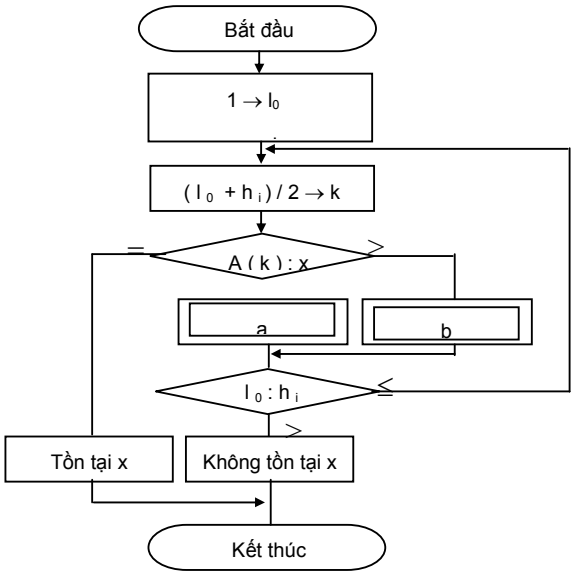
Vì vị trí cuối hoặc không được kiểm tra trong bước 5, ô trống sẽ là “giá trị được tìm thấy”, ví dụ X phù hợp với giá trị phần tử hiên tại. → Câu trả lời là d

Q3

Trả lời

	a	b
C	$k + 1 \rightarrow lo$	$k - 1 \rightarrow hi$

Mô tả



Phương pháp tìm kiếm nhị phân là phương pháp thu hẹp dữ liệu đích trong khi chia vùng tìm kiếm thành hai phần.

1) chỗ trống a

Nếu A(k) < X, chia vùng có nửa cao hơn thành hai phần, vì vậy "k+1 → lo".

2) khoảng trống b

Nếu A(k) > X, chia vùng nửa thấp hơn thành hai phần, vì vậy k-1 → hi

Do đó, câu trả lời là c.

Q4

Trả lời

b. 11

Mô tả

Trong tìm kiếm nhị phân, số lần so sánh trung trình và lớn nhất với số các phần tử là N:

-Số lần so sánh trung bình là $[\log_2 N]$

-Số lần so sánh lớn nhất là $[\log_2 N] + 1$

([ ] là ký tự Gaussian và số thập phân của giá trị đưa ra trong ký tự được bỏ bớt.)

Số lần trung bình là k thì

$1 <= 2000/2^k < 2$

$2^k <= 2000/2^k < 2^{k+1}$

Ở đây,

$\log_2 1024 = \log_2 (2)^{10} = 10 <= \log_2 2000 < \log_2 2048 = \log_2 (2)^{11} = 11$

Vì vậy k = 10 và số lần lớn nhất là k+1 = 11

→ câu trả lời là b.

Q5

Trả lời

d. Nếu số dữ liệu là 10 hay nhỏ hơn, thì số lần so sánh trung bình mà phương pháp duyệt tuyến tính đòi hỏi là nhỏ hơn số lần trung bình của phương pháp duyệt nhị phân.

Mô tả

Trong tìm kiếm nhị phân, số lần so sánh trung bình và lớn nhất với số các phần tử là N:

-Số lần trung bình là $[\log_2 N]$

-Số lần lớn nhất là $[\log_2 N] + 1$

Trong tìm kiếm tuyến tính, số lần so sánh lớn nhất là: Nếu số phần tử là N,

-Số lần so sánh trung bình là N/2

-Số lần so sánh lớn nhất là N

a. Dùng phương pháp duyệt nhị phân, dữ liệu phải được sắp xếp.

a là đúng.

b. Để duyệt 100 dữ liệu bằng việc dùng phương pháp duyệt nhị phân, số lần so sánh tối đa được cần tới để tìm ra dữ liệu đích là 7.

Nếu N=100, số lần trung bình cho tìm kiếm nhị phân là

$\log_2 100 < \log_2 128 = \log_2 (2)^7$

Vì vậy trung bình là 6 và lớn nhất là 7 → b cũng đúng.

c. Nếu phương pháp duyệt tuyến tính được dùng, số lần so sánh không nhất thiết giảm đi cho dù dữ liệu đã được lưu giữ.

C cũng đúng.

d. Nếu số dữ liệu là 10 hay nhỏ hơn, thì số lần so sánh trung bình mà phương pháp duyệt tuyến tính đòi hỏi là nhỏ hơn số lần trung bình của phương pháp duyệt nhị phân.

Nếu N=10,

Số lần tìm kiếm tuyến tính trung bình là 10/2 = 5

Số lần tìm kiếm nhị phân trung bình là $\log_2 10 < \log_2 (2)^4 = 4$

Vì vậy d là sai → Đây là câu trả lời

e. Nếu số dữ liệu tăng lên từ 100 tới 1,000, thì số lần so sánh tăng lên 10 lần hơn phương pháp duyệt tuyến tính được dùng. Bằng việc dùng phương pháp duyệt nhị phân, số này tăng lên hai hay ít hơn.

Số lần trung bình của tìm kiếm nhị phân với N=100 và N=1000 là như dưới đây. Vì vậy e cũng đúng.

Nếu N=100, $\log_2 100 < \log_2 128 = \log_2 (2)^7$

Nếu N=1000, $\log_2 1000 < \log_2 1024 = \log_2 (2)^{10}$

Q6

Trả lời

	a	b	c	d
D	Tăng dần	Sắp xếp	Xắp xếp ngoài	Gộp

Mô tả

1) a, b

“Sắp xếp dữ liệu theo thứ tự tuần tự từ giá trị nhỏ nhất đến giá trị lớn nhất” nghĩa là

“sắp xếp tăng dần”.

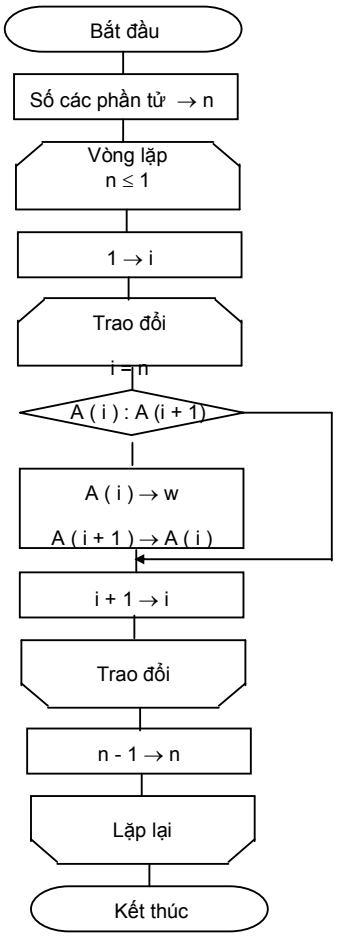
2) c
Nếu dãy dữ liệu đích là lưu trữ phụ, thì hoạt động này gọi là “sắp xếp ngoài”.
c.f. Sắp xếp ngoài nghĩa là sắp xếp dữ liệu trong đơn vị bộ nhớ chính. Sắp xếp ngoài nghĩa là dữ liệu lưu trữ được lưu trữ trên đĩa từ hoặc đơn vị lưu trữ phụ khác.
3) d
Tích hợp hai hoặc nhiều tệp được lưu trữ theo thứ tự trong một tệp được gọi là “gộp”

Q7

Trả lời

e. Sắp xếp nổi bọt

Mô tả



Trong lưu đồ trên, so sánh A(i) và A(i+1), nếu A(i) > A(i+1), chúng được đổi chỗ, ví dụ, một cặp phần tử tiếp theo mỗi phần tử trong chuỗi được so sánh và nếu chuỗi số là sai thì đổi chỗ.
Phần này mô tả sắp xếp nổi bọt. → câu trả lời là e.

Q8

Trả lời

e. 10,000

Mô tả

Trong cơ cấu sắp xếp nổi bọt, hai phần tử tiếp theo mỗi phần tử trong chuỗi được so sánh và nếu chuỗi sai thì đổi chỗ.
Sự phức tạp của máy điện toán là n^2 (tương ứng với một hình vuông có cạnh n, n: số dữ liệu)
Nếu số dữ liệu lớn gấp 100 lần (trước kia là 1000 và bây giờ là 100 000), thì thời gian ước tính sẽ là $(100)^2$ = 10,000 lần so với trước kia.
→ câu trả lời là e. 10,000.

Q9

Trả lời

d. Sắp xếp vun đống là phương pháp sắp xếp dữ liệu bằng việc biểu diễn một vùng chưa sắp xếp như một cây con, lấy giá trị tối đa hay tối thiểu từ miền chưa sắp, chuyển giá trị tối đa hay tối thiểu vào vùng được sắp xếp và lặp lại việc này để thu hẹp dần miền chưa sắp xếp.

Mô tả

a. Sắp xếp nhanh Quick sort là phương pháp sắp xếp dữ liệu theo các dãy con bao gồm các khoản mục dữ liệu được lấy từ các khoảng và sắp xếp các dãy con nhỏ hơn bao gồm các khoản mục dữ liệu được lấy tại khoảng nhỏ nhất.

Câu này mô tả “phương pháp sắp xếp bóc vỏ” mà hai yếu tố dữ liệu được xác định ngay từ mỗi yếu tố tại khoảng thời gian nào đó được chọn ra từ dãy dữ liệu.

b. Sắp xếp bóc vỏ Shell sort là phương pháp sắp xếp dữ liệu bằng cách so sánh một cặp các phần tử kề nhau và tráo đổi chúng nếu phần tử thứ hai lớn hơn phần tử thứ nhất.

Câu này mô tả “phương pháp sắp xếp nổi bọt” ,là phương pháp so sánh một cặp các phần tử kề nhau.

c. Sắp xếp nổi bọt Bubble là phương pháp sắp xếp dữ liệu bằng cách đặt một giá trị tham chiếu trung gian, phân bổ các phần tử với giá trị lớn hơn giá trị tham chiếu trong phần này và đặt các phần tử với giá trị nhỏ hơn giá trị tham chiếu vào phần kia và lặp lại việc

này cho từng phần riêng một.

Câu này mô tả “phương pháp sắp xếp nhanh”.

d. Sắp xếp vun đống là phương pháp sắp xếp dữ liệu bằng việc biểu diễn một vùng chưa sắp xếp như một cây con, lấy giá trị tối đa hay tối thiểu từ miền chưa sắp xếp, chuyển giá trị tối đa hay tối thiểu vào vùng được sắp xếp và lặp lại việc này để thu hẹp dần miền chưa sắp xếp.

Đây là mô tả về phương pháp sắp xếp vun đống. → Đây là câu trả lời

Q10

Trả lời

c. Dữ liệu được chia thành một nhóm các dữ liệu nhỏ hơn một giá trị tham chiếu và nhóm kia là các dữ liệu lớn hơn giá trị tham chiếu. Trong từng nhóm, một giá trị tham chiếu mới được lựa ra và dữ liệu giống thế lại được phân chia thành hai nhóm dựa trên giá trị tham chiếu này. Việc này được thực hiện lặp lại.

Mô tả

Trong câu hỏi này cần tìm mô tả thích hợp về sắp xếp nhanh.

Phương pháp sắp xếp nhanh do Hoare thiết kế. Nó diễn tả phương pháp sắp xếp nhanh nhất sử dụng phương pháp đệ quy.

Giá trị tham chiếu (điểm thử hoặc giá trị thử) được chọn từ dữ liệu được sắp xếp.

Giá trị trung bình của ba phần tử (phần tử trái, chính giữa, phải) được sử dụng thường xuyên. Sau đó, phần tử dữ liệu nhỏ hơn giá trị chính được chuyển sang trái của giá trị chính trong khi những phần tử dữ liệu lớn hơn giá trị chính được chuyển sang phải. Tất cả phần tử dữ liệu được chia thành hai phần.

Hoạt động phân chia này được thực hiện lặp lại cho đến khi chỉ còn lại một phần tử.

→ c. là mô tả thích hợp về phương pháp sắp xếp nhanh → đây là câu trả lời

a. So sánh và tráo đổi được thực hiện cho hai dữ liệu xa nhau với khoảng cách nào đó. Khoảng cách này dần dần và liên tục được thu hẹp để sắp xếp mọi dữ liệu.

a. mô tả phương pháp sắp xếp nhanh.

b. Giá trị tối thiểu thứ nhất được tìm ra trong dữ liệu. Giá trị tối thiểu thứ hai được tìm ra trong dữ liệu mà trong đó giá trị tối thiểu thứ nhất không được bao hàm. Việc này được thực hiện lặp lại.

b. mô tả phương pháp lựa chọn cơ bản.

d. Dữ liệu kề nhau được so sánh và tráo đổi lặp đi lặp lại để cho phép dữ liệu nhỏ hơn được chuyển về cuối mảng dữ liệu.

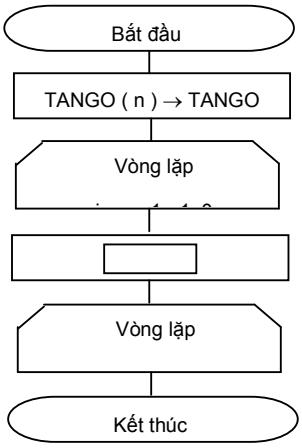
d. mô tả phương pháp sắp xếp nổi bọt.

Q11

Trả lời

a. TANGO (i) → TANGO (i+1)

Mô tả



Thuật toán này bao gồm

1) TANGO(n) → TANGO(0)

2) Lặp, giá trị chỉ số "I" thay đổi từ n-1 tới 0 với lượng giảm bằng 1

Áp dụng thuật toán trên ta có.

Giả sử n=5 và mảng "TANGO" là ban đầu. Ví dụ các từ "FE", "SW", "JITEC", "JIPDEC" và "METI" được lưu trữ lần lượt theo thứ tự trong TANGO(1), TANGO(2), TANGO(3), TANGO(4) và TANGO(5).

	FE	SW	JITEC	JIPDEC	METI
TANGO(0)	TANGO(1)	TANGO(2)	TANGO(3)	TANGO(4)	TANGO(5)

Sau khi áp dụng thuật toán này, từ trong TANGO(n) giả sử được lưu trữ trong TANGO(1), sau đó các từ còn lại được dịch sang phải.

METI	METI	FE	SW	JITEC	JIPDEC
TANGO(0)	TANGO(1)	TANGO(2)	TANGO(3)	TANGO(4)	TANGO(5)

Sau khi thực hiện "TANGO(5)→TANGO(0)", thực hiện theo vòng lặp (giá trị chỉ số "i" thay đổi từ n-1 đến 0 với lượng giảm một giá trị)

TANGO(4) → TANGO(5)

TANGO(3) → TANGO(4)

:

TANGO(0) → TANGO(1)

Điều này có thể được mô tả sử dụng chỉ số "I" như TANGO(i) → TANGO(i+1).

Vì vậy câu trả lời là a.

b. TANGO (i) → TANGO (n-i)
c. TANGO (i+1) → TANGO (n-i)
d. TANGO (n-i) → TANGO (i)

b,c,d là sai.

Q12

Trả lời

b. Như được thấy theo quan điểm hình học, phương pháp thu lấy lời giải xấp xỉ bằng việc dùng đường tiếp tuyến của y = f(x).

Mô tả

Phương pháp thuật toán Newton được mô tả như sau. Như được chỉ ra dưới đây, nó thu được lời giải bằng cách sử dụng đường tiếp tuyến y=f(x). Vì vậy câu trả lời là b.

Bước 1

Vẽ đường tiếp tuyến y = f(x) tại p1(x1, y1) và thu được điểm x2 là giao của đường tiếp tuyến và trục x.

Bước 2

Vẽ đường tiếp tuyến y = f(x) tại p2 (x2, y2) và thu được điểm x1 là giao của đường tiếp tuyến và trục x. Khi bước này được thực hiện lặp đi lặp lại thì đường tiếp tuyến chuyển gần tới kết quả.

Bước 3

Sự khác nhau giữa các giá trị xấp xỉ gần kề nhau thu được trong bước 2 được so sánh với độ chính xác của giá trị hội tụ được xác định trước. Thực hiện lặp lại bước 1 và 2 cho tới khi sự khác nhau này nhỏ hơn giá trị hội tụ được xác định trước.

Biểu thức đường tiếp tuyến tại p1 là $y-f(x_1) = f'(x_1)(x-x_1)$, điểm x_2 là giao của đường tiếp tuyến và trục x sử dụng biểu thức sau:

$$x_{i+1} = x_1 - \frac{f(x_i)}{f'(x_i)} \quad (i = 0, 1, 2, ...)$$

a. Mặc dầu một hàm f(x) không thể được lấy vi phân, nhưng vẫn có thể thu được một lời giải xấp xỉ.

Điều này là sai vì phương pháp này sử dụng vi phân như giải thích ở trên.

c. Cung cấp hai giá trị khởi đầu khác nhau.

Điều này cũng sai vì chỉ một kết quả được đưa ra từ một giá trị ban đầu.

d. Với bất kì giá trị ban đầu nào bao giờ cũng thu được một giá trị xấp xỉ.

Trong trường hợp $f'(x_n) = 0$, độ hội tụ sẽ không bao giờ diễn ra, qua đó không thu được giá trị xấp xỉ.

Trả lời cho Quyển 3 Chương 3 (Thiết kế trong)

Danh sách đáp án

Đáp án									
Q 1:	b	Q 2:	a,e	Q 3:	a	Q 4:	e	Q 5:	d

Trả lời và mô tả

Q1

Trả lời

b. Thiết kế dữ liệu vật lý

Mô tả

Trong câu hỏi này cần tìm công việc thực hiện thích hợp nhất trong thiết kế trong, như bộ phận của hoạt động phát triển hệ thống trong số các lựa chọn sau.

a. Thiết kế mã b. Thiết kế dữ liệu vật lý c. Thiết kế chương trình theo cấu trúc
d. Định nghĩa các yêu cầu e. Thiết kế dữ liệu Logic

Thiết kế trong, không giống như thiết kế ngoài, xác định cách thực hiện hệ thống từ quan điểm của nhà phát triển.

Các chức năng của hệ thống được chia theo module (các chương trình) và giao diện giữa các module (các chương trình) được định nghĩa rõ ràng.

Thiết kế trong dựa trên kết quả của thiết kế ngoài, kết quả thiết kế trong được sử dụng trong bước tiếp theo, thiết kế chương trình.

Các hoạt động của thiết kế trong bao gồm

- Phân tích chức năng
- Thiết kế cấu trúc
- Thiết kế dữ liệu vật lý
- Thiết kế vào ra
- Xem xét thiết kế trong

Vì vậy câu trả lời là b. Thiết kế dữ liệu vật lý

Q2

Trả lời

a. Ước lượng dung lượng và thời gian truy nhập e. Xác định cách bố trí bản ghi

Mô tả

Trong câu hỏi này cần xác định hai tác vụ được thực hiện trong thiết kế dữ liệu vật lý trong giai đoạn thiết kế trong.

a. Ước lượng dung lượng và thời gian truy nhập b. Xác định các yếu tố dữ liệu
c. Phân tích các mối quan hệ dữ liệu d. Tạo các đặc tả tệp
e. Xác định cách bố trí bản ghi

Thiết kế dữ liệu vật lý bao gồm các hoạt động như giới hạn dung lượng tệp, thiết kế lưu trữ cơ sở dữ liệu, v.v…

Trong số các lựa chọn ở trên, a và e được thực hiện trong thiết kế trong.

Các lựa chọn khác (b, c và d) được thực hiện trong thiết kế ngoài.

Q3

Trả lời

a DFD

Mô tả

Trong câu hỏi này cần xác định biểu đồ được sử dụng trong phân tích cấu trúc để diễn tả các chức năng và luồng dữ liệu bằng cách sử dụng các ký hiệu đưa ra luồng dữ liệu, xử lý (các chức năng), kho dữ liệu và nguồn ngoài (nguồn dữ liệu được sinh ra và gửi đi).

a. DFD b. ERD c. Sơ đồ NS d. Biểu đồ chuyển trạng thái
e. Biểu đò Warnier

a DFD

Chuẩn DFD cho “biểu đồ luồng dữ liệu: và biểu đồ này mô tả trong câu hỏi. → Đây là câu trả lời

b Biểu đồ E-R

Biểu đồ này được sử dụng trong mô hình dữ liệu để đưa ra các thực thể và mối quan hệ giữa chúng.

c Sơ đồ NS

Biểu đồ này được sử dụng trong lập trình cấu trúc.

d Biểu đồ chuyển trạng thái

Biểu đồ này diễn tả các trạng thái và các giao dịch giữa các trạng thái.

e Biểu đồ Warnier

Biểu đồ này cũng được sử dụng trong thiết kế module.

Q4

Trả lời

e. Mối quan hệ giữa các bước vào/ra và xử lí có thể được biểu diễn rõ ràng.

Mô tả

Trong câu hỏi này cần xác định mô tả thích hợp về HIPO, một trong các phương pháp thiết kế cấu trúc.

a. Biểu đồ các nội dung và biểu đồ luồng dữ liệu được dùng.
b. Thông tin điều khiển được truyền qua giữa các khối xử lí được mô tả cùng với các mũi tên trong biểu đồ nội dung.
c. Biểu đồ nội dung chỉ ra chức năng toàn thể của chương trình, và các số được đưa vào trong các khối xử lí chỉ ra trình tự xử lí.
d. Các kí hiệu trong lưu đồ được dùng để chỉ ra cái gì được chọn và cái gì được lặp lại.
e. Mối quan hệ giữa các bước vào/ra và xử lí có thể được biểu diễn rõ ràng.

Chuẩn HIPO cho “biểu đồ phân cấp cộng cái vào với xử lý cái ra”
Phần “cấp bậc” là bảng hiển thị nội dung mà hiển thị các module trong cấp bậc giống nhau tới biểu đồ tổ chức. Phần “cộng cái vào xử lý cái ra” là biểu đồ đưa ra tất cả các quy trình, đầu vào và đầu ra.
Vì vậy a,b,c và d đều đúng.
Câu trả lời là e.

Q5

Trả lời

d. Để hoàn thành tiến trình đang diễn ra, màn hình phải được thiết kế để ngăn cản người dùng bỏ dở việc đưa dữ liệu vào, hay trở lại màn hình trước đó.

Mô tả

Trong câu hỏi này cần xác định xem xét không thích hợp về thiết kế màn hình trong thiết kế trong và thiết kế ngoài.

a. Việc chuyển màn hình nên được thiết kế với việc xem xét không chỉ việc lựa từng bước bằng việc dùng một menu, mà còn với truy nhập trực tiếp vào màn hình mong muốn cho người dùng đã thành thạo.
b. Từng khoản mục mà dữ liệu được đưa vào trên màn hình phải được bao trong ngoặc vuông, để nhấn mạnh rằng nó là trường đưa vào dữ liệu.
c. Cách bố trí màn hình phải được thiết kế theo cách các khoản mục cần tham chiếu được thu xếp từ trái sang phải, và từ trên xuống dưới.
d. Để hoàn thành tiến trình đang diễn ra, màn hình phải được thiết kế để ngăn cản người dùng bỏ dở việc đưa dữ liệu vào, hay trở lại màn hình trước đó.
e. Bố trí màn hình phải được chuẩn hoá; các qui tắc về vị trí cho hiển thị tiêu đề và thông

báo phải được thiết lập.

Trong các lựa chọn trên, d là không thích hợp vì nó chỉ thu hút tới sự thuận tiện của hệ thống mà không thuận tiện cho người dùng hoặc giảm hoạt động. Thiết kế màn hình nên được thực hiện bằng cách chú ý tới yếu tố giao diện con người.

Trả lời cho Quyển 3 Chương 4 (Thiết kế chương trình)

Danh sách đáp án

Đáp án

Q 1:	e	Q 2:	c	Q 3:	c	Q 4:	d	Q 5:	c
Q 6:	a	Q 7:	d	Q 8:	b				

Trả lời và mô tả

Q1

Trả lời

e. Nên đưa vào các chú thích đúng để làm cho dễ hiểu logic bên trong mô đun.

Mô tả

Trong câu hỏi này cần xác định nhận xét không thích hợp trong phân hoạch module.

a. Số các mô đun cấp dưới mà một mô đun có thể gọi tới phải được giới hạn.
b. Mô đun phải được thiết kế sao cho nó chứa một số đúng các bước.
c. Trong thiết kế cấu trúc cấp bậc mô đun này gọi tới mô đun kia, phải để ý tới việc giữ chiều sâu trong giới hạn xác định.
d. Giao diện giữa các mô đun phải được làm đơn giản hoá.
e. Nên đưa vào các chú thích đúng để làm cho dễ hiểu logic bên trong mô đun.

Tất cả a, b, c và d là xem xét thiết kế module.

e là thực hành mã đúng nhưng không liên quan tới phân hoạch module.

Vì vậy, câu trả lời là e.

Q2

Trả lời

	Chức năng			
	Vào dữ liệu	Chọn số	Tính giá trị trung bình	Hiển thị kết quả
c	Nguồn	Nguồn	Biến đổi	Bể chứa

Mô tả

Trong câu hỏi này, kết hợp đúng các đặc điểm chức năng của chương trình dựa trên phân hoạch STS. Chương trình này đọc dữ liệu, chọn dữ liệu số và đưa ra giá trị trung bình.

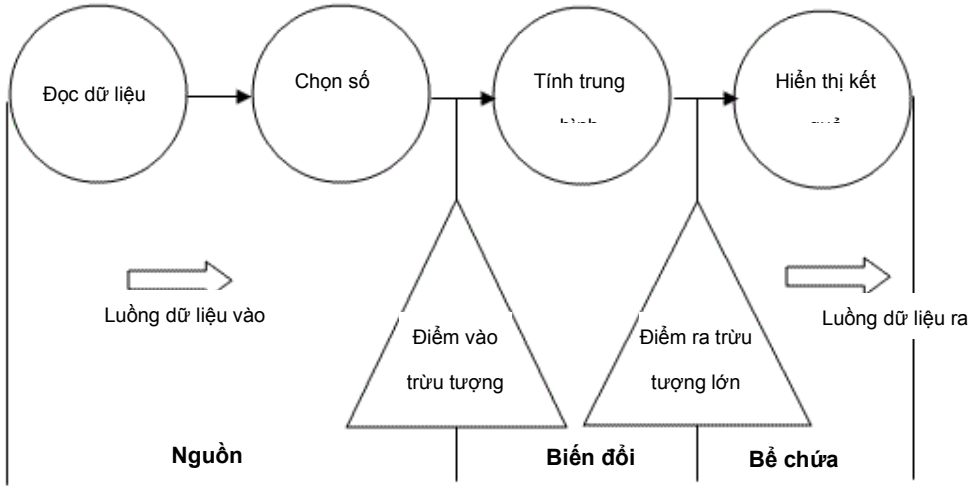
	Chức năng			
	Vào dữ liệu	Chọn số	Tính giá trị trung bình	Hiển thị kết quả
a	Bể chứa	Bể chứa	Nguồn	Biến đổi
b	Bể chứa	Nguồn	Nguồn	Biến đổi
c	Nguồn	Nguồn	Biến đổi	Bể chứa
d	Nguồn	Biến đổi	Biến đổi	Bể chứa
e	Biến đổi	Bể chứa	Bể chứa	Nguồn

Trong phương pháp phân hoạch STS (hoặc phân tích), một chương trình được chia thành ba phần như sau

1) Nguồn (Chức năng xử lý đầu vào)
2) Biến đổi (Chức năng xử lý dữ liệu)
3) Bể chứa (Chức năng xử lý đầu ra)

Trong phương pháp này, điểm vào trừu tượng hóa lớn nhất, nơi dữ liệu không thể được xem xét lâu hơn dữ liệu đầu vào và điểm ra trừu tượng lớn nhất, nơi dữ liệu bắt đầu mang hình như dữ liệu đầu ra.

Chương trình trong câu hỏi “đọc dữ liệu”, “chọn dữ liệu số” và “hiển thị giá trị trung bình”. Hơn nữa “tính dữ liệu” cũng được thực hiện. Điều này cộng tính thực tế trung bìnhcó thể được đưa ra như sau.



Q3

Trả lời

c. Phương pháp phân hoạch giao tác

Mô tả

Trong câu hỏi này cần xác định tên phương pháp phân tích được dùng cho sự chia theo chương trình cập nhật tệp.

● ● ● Cập nhật tệp● Cập nhật trả tiền cơ bản● ● Cập nhật công nhật● ● Cập nhật khấu trừ● ●

a. Phương pháp phân hoạch STS b. Phương pháp Jackson

c. Phương pháp phân hoạch giao tác d. Phương pháp Warnier

Phân tích là tiến trình chia chức năng thành module thành các mẩu ít phức tạp nhất. Có ba loại phương pháp phân tích "hướng luồng dữ liệu".

1) Phân tích bể chứa/Biến đổi/Nguồn
2) Phân tích giao tác
3) Phân tích chức năng

Có các phương pháp phân tích "hướng cấu trúc dữ liệu" sau:

1) Phương pháp Jackson
2) Phương pháp Warnier

Trong trường hợp trên, khi mỗi phần sau khi phân tích diễn tả một giao tác khác nhau, câu trả lời là c. Phân tích giao tác.

Q4

Trả lời

d. Phương pháp phân hoạch giao tác

Mô tả

Trong câu hỏi này cần tìm phương pháp sử dụng để chuyển biểu đồ luồng dữ liệu được tạo ra bởi phân tích cấu trúc thành biểu đồ cấu trúc được sử dụng cho thiết kế cấu trúc?

a. Phương pháp KJ b. Phương pháp OMT c. Phương pháp Jackson d. Phương pháp phân hoạch giao tác

Câu trả lời là d.

a là tiếp cận khoa học để giải quyết vấn đề, do Jiro Kawakita ở Nhật Bản phát triển

b là một trong các phương pháp hướng đối tượng.

c là phương pháp phân hoạch hướng cấu trúc dữ liệu.

Q5

Trả lời

c. Phương pháp Jackson

Mô tả

Trong câu hỏi này cần xác định kỹ thuật phân hoạch module tập trung vào cấu trúc dữ liệu.

a. Phương pháp phân hoạch chức năng chung

b. Phương pháp phân hoạch nguồn/biến đổi/bể chứa (phương pháp phân hoạch STS)

c. Phương pháp Jackson

d. Phương pháp phân hoạch giao tác (phương pháp phân hoạch TR)

Tất cả a, b và d là kỹ thuật phân hoạch hướng luồng dữ liệu.

Vì vậy câu trả lời là c, phương pháp Jackson hoặc JSP (Jackson Structured Programming.)

Q6

Trả lời

a. Biểu đồ cấu trúc của dữ liệu vào và ra được vẽ với chú ý chính dồn vào cấu trúc dữ liệu. Biểu đồ cấu trúc chương trình được chuẩn bị dựa trên biểu đồ cấu trúc dữ liệu vào/ra.

Mô tả

Trong câu hỏi này cần xác định mô tả đúng nhất về phương pháp Warnier được sử dụng để tạo thết kế chương trình theo cấu trúc.

a. Biểu đồ cấu trúc của dữ liệu vào và ra được vẽ với chú ý chính dồn vào cấu trúc dữ liệu. Biểu đồ cấu trúc chương trình được chuẩn bị dựa trên biểu đồ cấu trúc dữ liệu vào/ra.

b. Các chức năng trong luồng dữ liệu được gộp nhóm vào trong các loại nguồn, biến đổi và bể chứa với chú ý chính được dồn vào luồng dữ liệu cần giải quyết.

c. Phần mềm được coi như một tuyển tập các dữ liệu và qui trình. Tính độc lập mô đun được tăng lên bằng cách bao bọc những dữ liệu và qui trình này.

d. Với chú ý chính được dồn vào cấu trúc điều khiển của chương trình, logic chương trình được thiết kế dựa trên luồng điều khiển, biểu lộ mối quan hệ gọi nhau.

Vì phương pháp Warnier là phương pháp phân hoạch hướng cấu trúc dữ liệu, nên câu trả lời là a.

Q7

Trả lời

d. Nối nội dung

Mô tả

Trong câu hỏi này cần xác định kiểu nối module mạnh nhất trong bốn lựa chọn sau:

a. Nối công cộng b. Nối dấu

c. Nối dữ liệu d. Nối nội dung

Nối module

Một trong những mục đích của thiết kế module là để tối thiểu hóa nối module với các module khác. Nói cách khác là tạo sự độc lập có thể. Có 7 loại nối module. Từ loại

có sự độc lập ít nhất tới loại có sự độc lập cao nhất như sau:

Không nối trực tiếp• Module không dùng chung dữ liệu với các module khác.• Tốt nhất• • Nối dữ liệu (c)• Hai module chỉ dùng chung thông tin qua phần tử dữ liệu đồng nhất.• • • Nối dấu (b)• Hai module tham chiếu cùng một cấu trúc dữ liệu không toàn cầu. • • • Nối điều khiển• Một module đi thẳng qua phần tử kiểm soát tới module khác • • • Nối ngoài• Một nhóm các module tham chiếu cấu trúc dữ liệu toàn cầu truy cập giới hạn (tệp toàn cầu). • • • Nối công công (a)• Nhóm các module tham chiếu cấu trúc dữ liệu toàn cầu. • • • Nối nội dung (d)• Một module tham chiếu trức tiếp vào trong các module khác • Tồi nhất hoặc mạnh nhất• •

Q8

Trả lời

b. a=3, b=7

Mô tả

Trong câu hỏi này cần xác định cặp chương trình sau đưa ra kết quả bao nhiêu

```
Main program Subprogram  sub (x, y)
a=3 ;    x=x+y ;
b=2 ;    y=x+y ;
sub (a, b) ;       return ;
```

x (đối chung) được gọi bởi giá trị và y được gọi bởi tham chiếu.

Trong chương trình chính, gán a =3 và b=2.
Sau đó chương trình con được thực hiện.
Vì x được gọi bởi giá trị và y được gọi bởi tham chiếu khi gọi chương trình con, nên giá trị của a được duyệt và địa chỉ của b được duyệt.
Vì vậy, khi gọi chương trình con, a không thay đổi, b trở thành 7.
(x=x+y=3+2=5, y=x+y=5+2=7)

Kết quả, a =3 (không thay đổi) và b=7. → Câu trả lời là b.

a. a=3, b=2 b. a=3, b=7 c. a=5, b=2 d. a=5, b=7

Trả lời cho Quyển 3 Chương 5 (Thực hiện chương trình)
Danh sách đáp án
Đáp án
Q 1:● c● Q 2:● d● Q 3:● b● Q 4:● d● Q 5:● a● ● Q 6:● a● Q 7:● b● ● ● ● ● ● ● ● ●
Trả lời và mô tả

Q1
Kiểm thử đơn vị
Trả lời
c. Kiểm thử đơn vị (Kiểm thử tích hợp (Kiểm thử hệ thống
Kiểm thử tích hợp
Mô tả
Trong câu hỏi này cần xác định trật tự kiểm thử thích hợp nhất.
Kiểm thử hệ thống
a. Kiểm thử hệ thống (kiểm thử tích hợp (kiểm thử đơn vị
b. Kiểm thử hệ thống (kiểm thử đơn vị (kiểm thử tích hợp
Kiểm thử vận hành
c. Kiểm thử đơn vị (kiểm thử tích hợp (kiểm thử hệ thống
d. Kiểm thử đơn vị (kiểm thử hệ thống (kiểm thử tích hợp

Thứ tự của kiểm thử như sau. (Câu trả lời là c.
◘◘◘◘ ‼ EMBED Word.Picture.8 ¶ ⊥

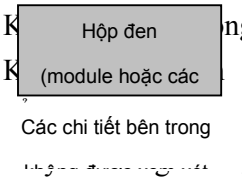
Q2
Trả lời
d. Kiểm thử hộp trắng
Mô tả
Trong câu hỏi này cần tìm thuật ngữ thích hợp nhất cho kiểm thử được tiến hành với sự chú ý nhất được dành cho cấu trúc bên trong của chương trình và thuật toán.
a. Kiểm thử hệ thống b. Kiểm thử trên xuống
c. Kiểm thử hộp đen d. Kiểm thử hộp trắng
e. Kiểm thử dưới lên
Các trường hợp kiểm thử cho kiểm thử hộp trắng được thiết kế bằng cách đưa ra xem xét đặc biệt tới cấu trúc các module trong và luồng điều khiển và logic.
Vì vậy, câu trả lời là d.

Q3
Trả lời
b. Kiểm thử hộp đen
Mô tả
Trong câu hỏi này cần tìm kỹ thuật để chuẩn bị kiểm thử dữ liệu và kiểm thử các chức năng của chương trình với sự chú ý nhất được dành cho các mối quan hệ giữa dữ liệu đầu vào và

kết quả đầu ra.

a K ng Kiểm thử hộp đen

c K Kiểm thử hộp trắng



Trong odule được xem như hộp đen. Sau đó, dữ liệu kiểm thử được thiết kế ch n (đầu ra và đầu vào) của mỗi module.

Vì vậy, câu trả lời là b.

EMBED Word.Picture.8 ¶ ⊥

Q4

Trả lời

d. Kiểm chứng rằng không có vấn đề gì với giao diện giữa các mô đun, là các cấu phần của chương trình

Mô tả

Trong câu hỏi này cần tìm mô tả thích hợp nhất về kiểm thử tích hợp được tiến hành trong tiến trình kiểm thử phát triển hệ thống, ngay sau khi một đơn vị kiểm thử (kiểm thử module) được hoàn thành.

Trong kiểm thử tích hợp, các kiểm thử được tiến hành để mỗi chương trình được đưa ra bằng cách liên kết các module. Các hoạt động của các chương trình và các giao diện giữa các module được kiểm tra.

a. Kiểm chứng rằng hệ thống có thể thực hiện không có lỗi nào cho tất cả các chức năng được xác định trong tài liệu thiết kế ngoài

b. Kiểm chứng rằng tập các mục tiêu về thời gian xử lí và mục tiêu thời gian đáp ứng đã được đạt tới

c. Kiểm chứng rằng không có vấn đề gì trong các kiểu và số thiết bị vào và ra và thiết bị truyền thông được ghép nối

d. Kiểm chứng rằng không có vấn đề gì với giao diện giữa các mô đun, là các cấu phần của chương trình

e. Kiểm chứng rằng việc cho chạy nhiều việc và ghép nối đồng thời các thiết bị cuối có thể được thực hiện như đã được xác định

a. Mô tả kiểm thử hệ thống

b. Mô tả kiểm thử hệ thống, đặc biệt là kiểm thử thực hiện

c. Mô tả kiểm thử vận hành

d. Mô tả kiểm thử tích hợp (Đây là câu trả lời

e. mô tả kiểm thử vận hành, đặc biệt là kiểm thử tải công việc

Q5

Trả lời

a. Kiểm thử được tiến hành bằng cách móc nối các mô đun theo trật tự mô đun thấp tới cao. Các khiển trình được cần tới làm cái thay thế cho các mô đun mức cao chưa hoàn tất.

Mô tả

Trong câu hỏi này cần xác định giải thích thích hợp hơn về kiểm thử dưới lên.

a. Kiểm thử được tiến hành bằng cách móc nối các mô đun theo trật tự mô đun thấp tới cao. Các khiển trình được cần tới làm cái thay thế cho các mô đun mức cao chưa hoàn tất.

b. Từng mô đun riêng lẻ được kiểm thử. Khi tất cả các mô đun đều đã được kiểm thử, thì chúng được móc nối và kiểm thử.

c. Kiểm thử được tiến hành bằng việc móc nối các mô đun theo thứ tự từ mô đun cao xuống mô đun thấp. Cuống được cần tới như cái thay thế cho các mô đun cấp thấp chưa hoàn tất.

d. Các kiểm thử được tiến hành theo trật tự kiểm thử đơn vị, tích hợp, hệ thống và vận hành.

Trong kiểm thử dưới lên, các module được tích hợp bằng cách di chuyển lên cấp thiết kế chương trình. (Câu trả lời là a.

b. Mô tả kiểm thử big bang.

c. Mô tả kiểm thử trên xuống.

d. Đưa ra thứ tự của kiểm thử (như được đưa ra trong câu 1)

Q6

Trả lời

a. Các trường hợp kiểm thử được chuẩn bị trước, và dữ liệu kiểm thử có thể đáp ứng các yêu cầu được xác định trong trường hợp kiểm thử được chuẩn bị.

Mô tả

Trong câu hỏi này cần tìm mô tả thích hợp hơn về dữ liệu kiểm thử được sử dụng để kiểm tra chương trình.

a. Các trường hợp kiểm thử được chuẩn bị trước, và dữ liệu kiểm thử có thể đáp ứng các yêu cầu được xác định trong trường hợp kiểm thử được chuẩn bị.

b. Chỉ dữ liệu kiểm thử có thể được xử lí đúng mới được chuẩn bị như các tiến trình kiểm thử.

c. Như dữ liệu được dùng cho kiểm thử, quãng 20% khối lượng dữ liệu cần xử lí trong các thao tác thực tế mới được chuẩn bị.

d. Dữ liệu kiểm thử bị bác bỏ như lỗi trong giai đoạn đưa vào không cần phải được cung cấp.

a. đúng (Đây là câu trả lời

b,d Các trường hợp kiểm thử có thể đưa ra cả hai điều kiện “đúng” và “sai” mà mỗi điều kiện tạo quyết định được thực hiện

Q7

Trả lời

b. Snapshot

Mô tả

Trong câu hỏi này cần xác định tên kỹ thuật gỡ lỗi để viết ra nội dung các biến hoặc thanh ghi

mỗi lần một câu lệnh đặc biệt được thực hiện.

a. Walk-through b. Snapshot

c. Bộ sinh dữ liệu kiểm thử d. Khiển trình

a Walk-through là một loại phương pháp xem xét

b Snapshot là sap chép tĩnh các chuyển động đặc biệt tại một thời điểm đặc biệt. Nó chứa các giá trị biến, giá trị thanh ghi và v.v... (Đây là câu trả lời

c Kỹ thuật này tự động sinh dữ liệu kiểm thử

d Kỹ thuật này thay thế module gọi được sử dụng trong kiểm thử dưới lên

Bảng đối chiếu thuật ngữ Anh - Việt

‼ INDEX \e " " \y \c "3" \h "[A]"¶

[A]• • • abstract data type• Kiểu dữ liệu trừu tượng• • approximation algorithm• Thuật toán xấp xỉ• • array type• Kiểu mảng• • ascending order• thứ tự tăng• • ASP• ASP• • assembler• Bộ hợp dịch• • • • • [B]• • • balanced tree• Cây cân bằng• • basic data structure• Cấu trúc dữ liệu cơ sở• • basic data type• Kiểu dữ liệu cơ sở• • basic exchange method• Phương pháp tráo đổi cơ sở• • basic insertion method• Phương pháp chèn cơ sở• • basic selection method• Phương pháp lựa cơ sở• • bi-directional list• Danh sách hai chiều• • big bang test• Kiểm thử Big-bang• • binary search method• Phương pháp duyệt nhị phân• • binary search tree• Cây tìm kiếm nhị phân• • binary tree• Cây nhị phân• • bisection method• Phương pháp phân đôi• • black box test• Kiểm thử hộp đen• • bottom-up programming• Lập trình dưới lên• • bottom-up test• Kiểm thử dưới lên• • Boyer-Moore method• Phương pháp Boyer-Moore• • branch• Nhánh• • breadth-first search method• Phương pháp duyệt chiều rộng trước• • B-tree• B-cây• • bubble chart• Sơ đồ bọt• • bubble sort• Sắp xếp nổi bọt• • • • • [C]• • • cell• Ô• • chain method• Phương pháp dây chuyền• • character string compression• Nén xâu kí tự• • character string processing• Xử lí xâu kí tự• • character string search• Duyệt xâu kí tự• • character type• Kiểu kí tự• • child• Con• • class library• Thư viện lớp• • coding rules (standards)• Những qui tắc viết mã (chuẩn)• • coincidental strength• Độ bền trùng hợp ngẫu nhiên• • collation algorithm• Thuật toán xếp bộ• • combination line• Đường tổ hợp• • combination test• Kiểm thử tổ hợp• • common coupling• Nối chung• • common function partitioning method• Phương pháp phân hoạch chức năng thường• • communicative strength• Độ bền trao đổi• • compiler• Bộ biên dịch• • computational complexity• Độ phức tạp tính toán• • content coupling• Nối nội dung• • control coupling• Nối điều khiển• • • • • [D]• • • DASD• DASD• • data check method• Phương pháp kiểm tra dữ liệu• • data coupling• Nối dữ liệu• • depth-first search method• Phương pháp duyệt chiều sâu trước• • dequeue• Lấy ra dữ liệu từ hàng đợi• • descending order• Sắp theo thứ tự giảm• • design review• Kiểm điểm thiết kế• • development tool• Công cụ phát triển• • DFD• Biểu đồ luồng dữ liệu• • dialog box• Hộp thoại• • Dijkstra search method• Phương pháp duyệt của Dijkstra• • direct access storage device• Thiết bị nhớ truy nhập trực tiếp• • direct organization file• Tệp tổ chức trực tiếp• • directed graph • Đồ thị có hướng• • directory• Danh mục• • divide-and-conquer method• Phương pháp chia và trị• • document• Tài liệu• • driver• Khiển trình• • dump routine• Trình xổ ra• • dynamic programming method• Phương pháp qui hoạch động• • • • • [E]• • • eight-queen question • Câu hỏi tám hậu• • encapsulation• Bao bọc• • enqueue• Lưu giữ dữ liệu vào hàng đợi • • enumeration type• Kiểu liệt kê• • ESDS• • • event-driven program• Chương trình được điều khiển theo biến cố• • exhaustive search method• Phương pháp duyệt vét cạn• • external coupling• Nối ngoài• • external sorting• Sắp xếp ngoài• • • • • [F]• • • FIFO• FIFO• • figure drawing• Vẽ hình• • file dump• Xổ tệp• • file processing• Xử lí tệp• • file updating• Cập nhật tệp• • first-in first-out• • • FIFO• • • fixed length record• Bản ghi chiều dài cố định• • flowchart• Lưu đồ• • functional programming• Lập trình hàm• • • • • [G]• • • gap• Lỗ hổng• • garbage• Rác• • graph• Đồ thị• • greedy algorithm method• Phương pháp thuật toán tham lam• • group control• Kiểm soát nhóm• • GUI• Giao diện người dùng đồ hoạ• • • • • [H]• • • hash• Băm• • hash method• Phương pháp băm• • heap• Đống• • HIPO• Hệ phân cấp cộng với cái vào xử lí cái ra• • home record• Bản ghi nhà• • • • • [I]• • • IDE• Môi trường phát triển tích hợp• • incremental test• Kiểm thử tăng dần• • index• Chỉ số• • index area• Vùng chỉ số• • indexed sequential file• Tệp tuần tự có chỉ số• • information hiding• Che giấu thông tin• • informational strength• Độ bền thông tin• • integer type• Kiểu nguyên• • integration test• Kiểm thử tích hợp• • interfaces between modules• Giao diện giữa các mô đun• • interpreter• Bộ thông

dịch• • • • • [J]• • • Jackson method • Phương pháp Jackson• • • • • [K]• • • knapsack problem• Bài toán ba lô• • KSDS• KSDS• • • • • [L]• • • language processor• Bộ xử lí ngôn ngữ• • last-in first-out• Vào sau ra trước• • LIFO• • • leaf• Lá• • LIFO• Vào-sau-ra-trước• • linear list• Danh sách tuyến tính• • linear search• Duyệt tuyến tính• • linear search method• Phương pháp duyệt tuyến tính• • list structure• Cấu trúc danh sách• • logic programming• Lập trình logic• • logical strength• Độ bền logic• • logical type• Kiểu logic• • • • • [M]• • • maximum abstraction input point• Điểm vào trừu tượng tối đa• • maximum abstraction output point• Điểm đưa ra trừu tượng tối đa• • MDI• Giao diện đa tài liệu • • member• Thành viên• • merge sort• Sắp xếp gộp• • module• Mô đun• • module coupling• Nối mô đun • • module independence• Tính độc lập của mô đun• • module logical design• Thiết kế logic mô đun• • module partitioning• Phân hoạch mô đun• • module strength• Độ bền mô đun• • Monte Carlo method• Phương pháp Monte Carlo• • multiway tree• Cây đa nhánh• • multi-window• Đa cửa sổ• • • • • [N]• • • N-ary tree• Cây N ngôi• • Newton's method• Phương pháp Newton• • node• Nút• • nonincremental test• Kiểm thử không tăng dần• • NULL• NULL• • numerical integration• Tích phân số• • • • • [O]• • • object-oriented programming• Lập trình hướng sự vật• • OCR• • • Bộ đọc kí tự quang học• • • one-dimensional array • Mảng một chiều• • operation test• Kiểm thử vận hành• • overflow area• Vùng tràn• • • • • [P]• • • parent• Cha mẹ• • partial type• Kiểu bộ phận• • partitioned organization file• Tệp tổ chức có phân hoạch• • peer-review• Kiểm điểm lại• • perfect binary tree• Cây nhị phân hoàn chỉnh• • physical data design• Thiết kế dữ liệu vật lí• • pivot• Thử• • pointer type• Kiểu con trỏ• • POP• Bật ra• • preprocessor • Bộ tiền xử lí• • primarity test problem• Bài toán kiểm thử số nguyên tố• • prime data area • Vùng dữ liệu chính• • probability algorithm• Thuật toán xác suất• • probability algorithm with bounded errors• Thuật toán xác suất với sai số bị chặn• • problem-oriented data structure• Cấu trúc dữ liệu hướng vấn đề• • procedural programming• Lập trình thủ tục• • procedural strength• Độ bền thủ tục• • process chart• Lưu đồ tiến trình• • program design document• Tài liệu thiết kế chương trình• • programming paradigm• Mô thức lập trình• • programming style• Phong cách lập trình• • PUSH• Ấn vào• • • • • [Q]• • • QC• • • quality control• Kiểm tra chất lượng• • queue• Hàng đợi• • quick sort• Sắp xếp nhanh• • • • • [R]• • • real number type• Kiểu số thực• • record type• Kiểu bản ghi• • recursive • Đệ qui• • recursive algorithm• Thuật toán đệ qui• • recursive call• Gọi đệ qui• • reduction method• Phương pháp rút gọn• • regression test• Kiểm thử rà lại• • reuse• Dùng lại• • ring list• Danh sách vòng• • root• Gốc• • RRDS• RRDS• • • • • [S]• • • sandwich test• Kiểm thử bánh mì kẹp thịt• • SDI• Giao diện một tài liệu• • segment• Đoạn• • sentinel search method• Phương pháp duyệt lính canh• • sequential method• Phương pháp tuần tự• • sequential organization file• Tệp tổ chức tuần tự• • Shaker sort• Sắp xếp sàng lắc• • Shell sort• Sắp xếp bóc vỏ• • short-cut key • Phím tắt• • shortest path problem• Bài toán đường đi ngắn nhất• • simple type• Kiểu đơn• • Simpson's method• Phương pháp Simpson• • spacing chart• Sơ đồ không gian• • SSP• SSP• • stable marriage problem• Bài toán hôn nhân ổn định• • stable matching• Đối sánh ổn định• • stack• Chồng• • stack pointer• Con trỏ chồng• • stamp coupling• Nối dấu• • state transition diagram• Biểu đồ chuyển trạng thái• • structured chart• Sơ đồ có cấu trúc• • structured design• Thiết kế có cấu trúc• • structured design method• Thương pháp thiết kế có cấu trúc• • structured programming• Lập trình có cấu trúc• • structured type• Kiểu có cấu trúc• • STS partitioning method• Phương pháp phân hoạch STS• • stub• Cuống• • subordinate module• Mô đun cấp dưới• • subprogram library• Thư viện chương trình con• • subscript• Chỉ số• • synonym• Đồng nghĩa• • synonym record• Bản ghi đồng nghĩa• • system test• Kiểm thử hệ thống• • • • • [T]• • • table• Bảng• • table search• Duyệt bảng• • test case design

manual• Tài liệu thiết kế trường hợp kiểm thử• • test data generation tool• Công cụ sinh dữ liệu kiểm thử• • three-dimensional array• Mảng ba chiều• • time strength• Độ bền thời gian• • top-down programming• Lập trình trên xuống• • top-down test• Kiểm thử trên xuống• • total test• Kiểm thử toàn diện• • TR partitioning method• Phương pháp phân hoạch TR• • tracer• Bộ dò vết• • trapezoidal rule• Qui tắc hình thang• • tree structure• Cấu trúc cây• • two-dimensional array• Mảng hai chiều• • • • • [U]• • • undefined length record • Bản ghi chiều dài không xác định• • undirected graph• Đồ thị vô hướng• • uni-directional list• Danh sách một chiều• • unit test• Kiểm thử đơn vị• • • • • [V]• • • validity• Tính hợp thức• • variable length record• Bản ghi chiều dài biến thiên• • virtual storage organization file• Tệp tổ chức lưu giữ ảo• • VSAM file• Tệp VSAM • • • • • [W]• • • Warnier method • Phương pháp Warnier• • waterfall model• Mô hình thác đổ• • web programming• Lập trình Web• • weighted graph• Đồ thị có trọng số• • white box test• Kiểm thử hộp trắng• • window• Cửa sổ• •
⊥ ‼ RD "../../../../No.3/No.3(Chap_1).doc" \f ⊥

Bảng đối chiếu thuật ngữ Việt - Anh

‼ INDEX \e " " \y \c "3" \h "[A]"¶

• ESDS• • • QC• • A• • • ấn vào‼ XE "ấn vào" ⊥ • PUSH• • ASP• ASP• • B• • • Bài toán ba lô• knapsack problem• • Bài toán đường đi ngắn nhất• shortest path problem• • Bài toán hôn nhân ổn định• stable marriage problem• • Bài toán kiểm thử số nguyên tố• primarity test problem• • Băm• hash• • Bản ghi chiều dài biến thiên• variable length record• • Bản ghi chiều dài cố định• fixed length record• • Bản ghi chiều dài không xác định• undefined length record• • Bản ghi đồng nghĩa• synonym record• • Bản ghi nhà• home record• • Bảng• table• • Bao bọc• encapsulation• • Bật ra• POP• • B-cây• B-tree• • Biểu đồ chuyển trạng thái• state transition diagram• • Biểu đồ luồng dữ liệu• DFD• • Bộ biên dịch• compiler• • Bộ dò vết• tracer• • Bộ đọc kí tự quang học• OCR• • Bộ hợp dịch• assembler• • Bộ thông dịch• interpreter• • Bộ tiền xử lí• preprocessor• • Bộ xử lí ngôn ngữ• language processor• • C• • • Cập nhật tệp• file updating• • Câu hỏi tám hậu• eight-queen question• • Cấu trúc cây• tree structure• • Cấu trúc danh sách• list structure• • Cấu trúc dữ liệu cơ sở• basic data structure• • Cấu trúc dữ liệu hướng vấn đề• problem-oriented data structure• • Cây cân bằng• balanced tree• • Cây N ngôi• N-ary tree• • Cây nhị phân• binary tree• • Cây nhị phân hoàn chỉnh• perfect binary tree• • Cây nhiều nhánh• multiway tree• • Cây tìm kiếm nhị phân• binary search tree• • Cha mẹ• parent• • Che giấu thông tin• information hiding• • Chỉ số• index• • Chỉ số• subscript• • Chương trình được điều khiển theo biến cố• event-driven program• • Con• child• • Con trỏ ngăn xếp• stack pointer• • Công cụ phát triển• development tool• • Công cụ sinh dữ liệu kiểm thử• test data generation tool• • Cửa sổ• window• • Cuống• stub• • D• • • Đa cửa sổ• multi-window• • Danh sách hai chiều• bi-directional list• • Danh sách một chiều• uni-directional list• • Danh sách tuyến tính• linear list• • Danh sách vòng• ring list• • DASD• DASD• • Đệ qui• recursive• • Điểm đưa ra trừu tượng tối đa• maximum abstraction output point• • Điểm vào trừu tượng tối đa• maximum abstraction input point• • Độ bền logic• logical strength• • Độ bền mô đun• module strength• • Độ bền thời gian• time strength• • Độ bền thông tin• informational strength• • Độ bền thủ tục• procedural strength• • Độ bền trao đổi• communicative strength• • Độ bền trùng hợp ngẫu nhiên• coincidental strength• • Độ phức tạp tính toán• computational complexity• • Đồ thị• graph• • Đồ thị có hướng• directed graph• • Đồ thị có trọng số• weighted graph• • Đồ thị vô hướng• undirected graph• • Đoạn• segment• • Đối sánh ổn định• stable matching• • Đống• heap• • Đồng nghĩa• synonym• • Dùng lại• reuse• • Đường tổ hợp• combination line• • Duyệt bảng• table search• • Duyệt tuyến tính• linear search• • Duyệt xâu kí tự• character string search• • F• • • FIFO• FIFO• • FIFO• first-in first-out• • G• • • Giao diện đa tài liệu • MDI• • giao diện giữa các mô đun• interfaces between modules• • Giao diện một tài liệu• SDI• • Giao diện người dùng đồ hoạ• GUI• • Gốc• root• • Gọi đệ qui• recursive call• • Hàng đợi• queue• • H• • • Hệ phân cấp cộng với cái vào xử lí cái ra• HIPO• • Hộp thoại• dialog box• • K• • • Khiển trình• driver• • Kiểm điểm lại• peer-review• • Kiểm điểm thiết kế• design review• • Kiểm soát nhóm• group control• • Kiểm thử bánh mì kẹp thịt• sandwich test• • Kiểm thử Big-bang• big bang test• • Kiểm thử đơn vị• unit test• • Kiểm thử dưới lên• bottom-up test• • Kiểm thử hệ thống• system test• • Kiểm thử hộp đen• black box test• • Kiểm thử hộp trắng• white box test• • Kiểm thử không tăng dần• nonincremental test• • Kiểm thử rà lại• regression test• • Kiểm thử tăng dần• incremental test• • Kiểm thử tích hợp• integration test• • Kiểm thử tổ hợp• combination test• • Kiểm thử toàn diện• total test• • Kiểm thử trên xuống• top-down test• • Kiểm thử vận hành• operation test• • Kiểm tra chất lượng• quality control• • Kiểu bản ghi• record type• • Kiểu bộ phận• partial type• • Kiểu có cấu trúc• structured type• • Kiểu con trỏ• pointer type• • Kiểu đơn• simple type• • Kiểu dữ liệu cơ sở• basic data type• • Kiểu dữ liệu trừu tượng• abstract data type• • Kiểu kí tự• character type• • Kiểu liệt kê• enumeration type• • Kiểu logic• logical type• • Kiểu mảng• array

type• • Kiểu nguyên• integer type• • Kiểu số thực• real number type• • KSDS• KSDS• • L• • • Lá• leaf• • Lập trình có cấu trúc• structured programming• • Lập trình dưới lên• bottom-up programming• • Lập trình hàm• functional programming• • Lập trình hướng sự vật• object-oriented programming• • Lập trình logic• logic programming• • Lập trình thủ tục• procedural programming• • Lập trình trên xuống• top-down programming• • Lập trình Web• web programming• • Lấy ra dữ liệu từ hàng đợi• dequeue• • LIFO• last-in first-out• • Lỗ hổng• gap• • Lưu đồ• flowchart• • Lưu đồ tiến trình• process chart• • Lưu giữ dữ liệu vào hàng đợi• enqueue• • M• • • Mảng ba chiều• three-dimensional array• • Mảng hai chiều• two-dimensional array• • Mảng một chiều• one-dimensional array • • Mô đun• module• • Mô đun cấp dưới• subordinate module• • Mô hình thác đổ• waterfall model• • Mô thức lập trình• programming paradigm• • Môi trường phát triển tích hợp• IDE• • Một danh mục• directory• • N• • • Nén xâu kí tự• character string compression• • Ngăn xếp• stack• • Nhánh• branch• • Những qui tắc viết mã (chuẩn)• coding rules (standards)• • Nối công cộng• common coupling• • Nối dấu• stamp coupling• • Nối điều khiển• control coupling• • Nối dữ liệu• data coupling• • Nối mô đun• module coupling• • Nối ngoài• external coupling• • Nối nội dung• content coupling• • NULL• NULL• • Nút• node• • O• • • Ô• cell• • P• • • Phân hoạch mô đun• module partitioning• • Phím tắt• short-cut key• • Phong cách lập trình• programming style• • Phương pháp băm• hash method• • Phương pháp Boyer-Moore• Boyer-Moore method• • Phương pháp chèn cơ sở• basic insertion method• • Phương pháp chia và trị• divide-and-conquer method• • Phương pháp dây chuyền• chain method• • Phương pháp duyệt chiều rộng trước• breadth-first search method• • Phương pháp duyệt chiều sâu trước• depth-first search method• • Phương pháp duyệt của Dijkstra• Dijkstra search method• • Phương pháp duyệt lính canh• sentinel search method• • Phương pháp duyệt nhị phân• binary search method• • Phương pháp duyệt tuyến tính• linear search method• • Phương pháp duyệt vét cạn• exhaustive search method• • Phương pháp Jackson• Jackson method• • Phương pháp kiểm tra dữ liệu• data check method• • Phương pháp lựa cơ sở• basic selection method• • Phương pháp Monte Carlo• Monte Carlo method• • Phương pháp Newton• Newton's method• • Phương pháp phân đôi• bisection method• • Phương pháp phân hoạch chức năng thường• common function partitioning method• • Phương pháp phân hoạch STS• STS partitioning method• • Phương pháp phân hoạch TR• TR partitioning method• • Phương pháp qui hoạch động• dynamic programming method• • Phương pháp rút gọn• reduction method• • Phương pháp Simpson• Simpson's method• • Phương pháp thiết kế có cấu trúc• structured design method• • Phương pháp thuật toán tham lam• greedy algorithm method• • Phương pháp tráo đổi cơ sở• basic exchange method• • Phương pháp tuần tự• sequential method• • Phương pháp Warnier• Warnier method• • Q• • • Qui tắc hình thang• trapezoidal rule• • R• • • Rác• garbage• • RRDS• RRDS• • S• • • Sắp theo thứ tự giảm • descending order• • Sắp xếp bóc vỏ• Shell sort• • Sắp xếp gộp• merge sort• • Sắp xếp ngoài• external sorting• • Sắp xếp nhanh• quick sort• • Sắp xếp nổi bọt• bubble sort• • Sắp xếp sàng lắc• Shaker sort• • Sơ đồ bọt• bubble chart• • Sơ đồ có cấu trúc• structured chart• • Sơ đồ không gian• spacing chart• • SSP• SSP• • T• • • Tài liệu• document• • Tài liệu thiết kế chương trình• program design document• • Tài liệu thiết kế trường hợp kiểm thử• test case design manual• • Tệp tổ chức có phân hoạch• partitioned organization file• • Tệp tổ chức lưu giữ ảo• virtual storage organization file• • Tệp tổ chức trực tiếp• direct organization file• • Tệp tổ chức tuần tự• sequential organization file• • Tệp tuần tự có chỉ số• indexed sequential file• • Tệp VSAM • VSAM file• • Thành viên• member• • Thiết bị nhớ truy nhập trực tiếp• direct access storage device• • Thiết kế có cấu trúc• structured design• • Thiết kế dữ liệu vật lí• physical data design• • Thiết kế logic mô đun• module

logical design• • Thử• pivot• • Thứ tự tăng dần• ascending order• • Thư viện chương trình con• subprogram library• • Thư viện lớp• class library• • Thuật toán đệ qui• recursive algorithm• • Thuật toán đối sánh• collation algorithm• • Thuật toán xác suất• probability algorithm• • Thuật toán xác suất với sai số bị chặn• probability algorithm with bounded errors• • Thuật toán xấp xỉ• approximation algorithm• • Tích phân số• numerical integration• • Tính độc lập của mô đun• module independence• • Tính hợp lệ• validity• • Trình xổ ra• dump routine• • V• • • Vào-sau-ra-trước• LIFO• • Vẽ hình• figure drawing• • Vùng chỉ số• index area• • Vùng dữ liệu chính• prime data area • • Vùng tràn• overflow area• • X• • • Xổ tệp• file dump• • Xử lí tệp• file processing• • Xử lí xâu kí tự• character string processing• • ⊥

Tra cứu thuật ngữ

‼ INDEX \e " " \c "2" \z "1033" ¶

⊥