CHƯƠNG 2
MẢNG VÀ DANH SÁCH

Ngô Công Thắng
Bộ môn Công nghệ phần mềm
Khoa Công nghệ thông tin
Website: dse.vnua.edu.vn/ncthang
Email: ncthang@vnua.edu.vn

1. Mảng

- Mảng là một tập hợp có thứ tự gồm một số cố định các phần tử cùng kiểu.
- Một phần tử mảng được chỉ ra bởi chỉ số, thể hiện thứ tự của phần tử trong mảng. Véc tơ là mảng 1 chiều có 1 chỉ số (i). Ma trận là mảng 2 chiều có 2 chỉ số (i, j). Không gian 3 chiều là mảng 3 chiều có 3 chỉ số. Không gian n chiều là mảng n chiều có n chỉ số.

1. Mảng

- Để truy nhập trực tiếp các phần tử, mảng chỉ dùng được cấu trúc lưu trữ kế tiếp.
- Có các phép tạo lập mảng, tìm kiếm 1 phần tử từ mảng, truy nhập một phần tử mảng.
- Không cho phép bổ sung hoặc loại bỏ một phần tử mảng.
- Mảng 2 chiều có m = 2 hàng, n = 3 cột. Tính chỉ số k truy nhập vào ô nhớ chứa phần tử a_{ij}.

4 5 9

7 10 1

4 5 9 7 10 1 => k = (i-1)*n + j

2. Danh sách

2.1. Khái niệm

- Danh sách là một tập hợp có thứ tự gồm một số biến động các phần tử cùng kiểu.
- Phép loại bỏ, bổ sung 1 phần tử là phép thường xuyên tác động lên danh sách.
- Ví dụ: Tập hợp người đến khám bệnh cho ta một danh sách. Người đến xếp hàng khám bổ sung ở phía sau, người được khám sẽ ra khỏi hàng (loại bỏ) ở phía trước.

2.1. Khái niệm

- Danh sách tuyến tính: Một danh sách mà quan hệ lận cận giữa các phần tử được xác định rõ ràng thì được gọi là danh sách tuyến tính. Véc tơ là một danh sách tuyến tính.
- Danh sách tuyến tính hoặc rỗng (không có phần tử nào) hoặc có dạng (a1, a2, ..., an) với ai , $1 \le i \le n$ là các phần tử.
- Trong danh sách tuyến tính tồn tại phần tử đầu là a1, phần tử cuối là an, phần tử thứ i là ai . Với ai bất kỳ $1 \le i \le n$ thì ai+1 gọi là phần tử sau ai ; $2 \le i \le n$ thì phần tử ai-1 là phần tử trước của ai .

2.2. Các phép toán trên danh sách

- Phép bổ sung: Có thể bổ sung phần tử vào danh sách.
- Phép loại bỏ: có thể loại bỏ một phàn tử ra khỏi danh sách.
- Phép ghép: có thể ghép hai hay nhiều danh sách thành một danh sách.
- Phép tách: có thể tách một danh sách thành nhiều danh sách.
- Phép cập nhật: cập nhật giá trị cho các phần tử của danh sách.

2.1. Khái niệm

- n là độ dài (kích thước) của danh sách, n có thể thay đổi.
- Một phần tử trong danh sách thường là một bản ghi (gồm một hoặc nhiều trường).
- Ví dụ 1: Danh mục điện thoại là một danh sách tuyến tính, mỗi phần tử của nó là một thuê bao gồm 3 trường: Họ tên chủ hộ, địa chỉ, số điện thoại.
- Ví dụ 2: Tệp(File) là một danh sách có kích thước lớn được lưu trữ ở bộ nhớ ngoài.

2.2. Các phép toán trên danh sách

- Phép sao chép: có thể sao chép một danh sách.
- Phép sắp xếp: Có thể sắp xếp các phần tử của danh sách theo một thứ tự nhất định.
- Phép tìm kiếm: Tìm kiếm trong danh sách một phần tử mà một trường nào đó có giá trị ấn định.

Ví dụ 1: Minh hoạ cho các phép toán trên danh sách được cài đặt trên mảng. Cho danh sách các số nguyên, thêm vào 1 số nguyên và loại bỏ một số nguyên.

2.3. Lưu trữ kế tiếp cho danh sách tuyến tính

- Dùng mảng một chiều làm cấu trúc lưu trữ danh sách tuyến tính. Tức là dùng vector lưu trữ (Vi) với 1≤ i ≤ m để lưu trữ một danh sách tuyến tính (a1,a2,...,an). Phần tử ai được chứa ở Vi .

a1 a2... ai... an

V1 V2 ... Vi ... Vn ... Vm

2.3. Lưu trữ kế tiếp cho danh sách tuyến tính

- Do số phần tử của danh sách tuyến tính luôn biến động, tức là kích thước n luôn thay đổi, do vậy m = max(n).
- Mặt khác, n không thể xác định chính xác mà chỉ dự đoán. Bởi vậy, nếu max(n) lớn sẽ lãng phí bộ nhớ cũng như lãng phí thời gian để thực hiện các thao tác để dồn các phần tử xuống khi ta thêm một phần tử vào danh sách tuyến tính.

3. Cấu trúc ngăn xếp (Stack)

3.1. Định nghĩa

- Stack là một kiểu danh sách tuyến tính đặc biệt mà phép bổ sung và phép loại bỏ luôn luôn thực hiện ở một đầu gọi là đỉnh (Top).
- Phép bổ sung và loại bỏ phần tử của ngăn xếp được thực hiện theo nguyên tắc 'Vào sau ra trước' (Last in - First out, viết tắt là LIFO).
- Stack có thể rỗng hoặc bao gồm một số phần tử.

3.2. Lưu trữ Stack bằng mảng

- Có thể lưu trữ Stack bởi một vector S gồm n ô nhớ kế tiếp nhau có chỉ số từ 1 đến n. Nếu T là chỉ số phần tử đỉnh của stack thì T sẽ có giá trị biến đổi khi stack hoạt động.
- Khi bổ sung 1 phần tử T sẽ tăng lên 1. Khi 1 phần tử bị loại khỏi stack thì T giảm đi 1.
- Ta quy ước khi stack rỗng T=0.

3.3. Các phép toán trên ngăn xếp

- Bổ sung một phần tử vào stack

- Vào: phần tử X, ngăn xếp (S,T)

- Ra: không có

{Thủ tục này bổ sung phần tử X vào ngăn xếp được lưu trữ bởi véc tơ S có kích thước là n, có chỉ số đỉnh là T.}

3.3. Các phép toán trên ngăn xếp

- Loại bỏ một phần tử ra khỏi stack

- Vào: Ngăn xếp (S, T)

- Ra: giá trị phần tử loại bỏ (đỉnh)

{Hàm này thực hiện việc loại bỏ phần tử ở đỉnh ngăn xếp (S,T) và trả về phần tử này.}

Thủ tục bổ sung một phần tử vào stack

```
Procedure  Push(S, T, X)
    1) Kiểm tra ngăn xếp đã đầy chưa?
        If  T = n   then  Begin   Write('Stack đã đầy')
                          Return
                    End;
    2) Thay đổi chỉ số
        T := T+1
    3) Bổ sung phần tử mới X
        S[T] := X
Return
```

Hàm loại bỏ phần tử khỏi ngăn xếp

```
Function  Pop(S,T)
    1) Kiểm tra xem stack có rỗng?
        If  T = 0  then   Begin
                          Write('Stack  rỗng')
                          Return;
                  End
    2) Tg := S[T];
    3) Thay đổi chỉ số đỉnh
        T := T-1
    4) Đưa phần tử bị loại ra
        Pop := Tg;
Return
```

Hàm trả về phần tử đỉnh ngăn xếp

```
Function  Top(S,T)
        1) {Kiểm tra xem stack có rỗng?}
                If  T = 0  then   Begin
                                        Write('Stack  rỗng')
                                        Return;
                                End
        2) {Trả về phần tử đỉnh}
                Top := S[T];
Return
```


Hàm kiểm tra ngăn xếp rỗng

```
Function  Empty(S,T)
        If  T = 0  then Empty:=TRUE
        Else Empty:=FALSE;
Return
```


Ví dụ về ứng dụng của ngăn xếp

- Viết giả mã có sử dụng ngăn xếp để đổi số nguyên hệ 10 sang hệ 2.
- Giải thuật: Lấy số hệ 10 chia nguyên liên tiếp cho 2, kết quả là phần dư của phép chia lấy theo thứ tự ngược lại. Áp dụng cơ chế vào sau ra trước, mỗi lần chia ta lấy số dư của phép chia đẩy vào stack (thủ tuc Push). Khi đã kết thúc phép chia kết quả lấy các số dư từ stack ra (hàm loại bỏ phần tử khỏi stack, Pop).

Ví dụ về ứng dụng của ngăn xếp

```
- Vào: n
- Ra: số nhị phân
Procedure  chuyen_doi (n);
  While  n<> 0   do  Begin
                        R:=n  mod 2
                        Call  Push(S,T,R);
                        n= n div 2
                        End;
  While  S<>NULL  do Begin
                R:=POP(S,T);  {lay R tu dinh T cua Stack S }
                        Write(R)
                End;
  Return;
```

Bài tập

- Ứng dụng ngăn xếp chuyển biểu thức trung tố hành hậu tố. Biết rằng biểu thức trung tố có dấu ngoặc đầy đủ.

4. Cấu trúc hàng đợi (Queue)

4.1. Định nghĩa

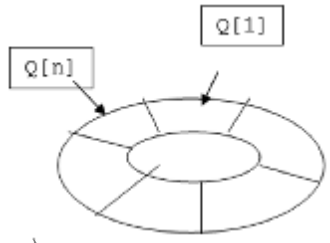
- Hàng đợi (queue) là kiểu danh sách tuyến tính mà phép bổ sung được thực hiện ở một đầu, gọi là lối sau (rear) và phép loại bỏ thực hiện ở một đầu khác, gọi là lối trước (front).
- Phép bổ sung và loại bỏ phần tử của hàng đợi được thực hiện theo nguyên tắc vào trước ra trước (First in - First out, viết tắt là FIFO).
- Hàng đợi có thể rỗng hoặc bao gồm một số phần tử.

4.2. Lưu trữ hàng đợi bằng mảng

- Dùng vector lưu trữ Q có n ô nhớ làm cấu trúc lưu trữ của hàng đợi, các ô nhớ có chỉ số từ 1 đến n.
- Để truy nhập vào hàng đợi ta dùng 2 biến chỉ số: R cho lối sau và F cho lối trước.
- Khi bổ sung 1 phần tử vào queue thì R sẽ tăng lên 1, còn khi loại bỏ một phần tử khỏi queue thì F tăng lên 1.
- Khi queue rỗng thì R=F=0.

4.2. Lưu trữ hàng đợi bằng mảng

- Các phần tử của queue sẽ di chuyển khắp không gian nhớ khi thực hiện phép bổ sung và loại bỏ.
- Khắc phục tình trạng đó người ta coi không gian nhớ tổ chức cho queue kiểu vòng tròn nghĩa là với vector lưu trữ Q thì phần tử Q[1] coi đứng trước Q[n]



4.3. Các phép toán trên Queue

a) Bổ sung một phần tử vào queue

- Vào: X, (Q, F, R)
- Ra: Không có

{Thủ tục này bổ sung phần tử X vào hàng đợi lưu trữ bởi vector Q có n ô nhớ theo kiểu vòng tròn, có biến chỉ số F, R.}

```
Procedure  CQINSERT (Q,F,R,X)
    1) {Kiểm tra đầy}
        If  (F=1)and(R=n) or (R+1=F) then  Begin
            Write( 'Hàng đợi đã đầy');
            Return
          End;
    2) {Thay đổi chỉ số R}
        If  R=0 Then F:=R:=1
        Else If R=n Then  R:=1
            Else R:= R+1;
    3. {Bổ sung X vào}
        Q[R]:=X;
Return  { kết thúc}
```


4.3. Các phép toán trên Queue

b) Loại bỏ phần tử ra khỏi queue

- Vào: Hàng đợi (Q,F,R)
- Ra: Trả về phần tử loại bỏ

{Hàm này loại bỏ phần tử ở lối trước của hàng đợi (Q,F,R) và trả về phần tử loại bỏ}

Thủ tục loại bỏ phần tử khỏi hàng đợi

```
Function CQDELETE(Q,F,R)
    1) {Kiểm tra rỗng}
        If  F=0  then  Begin
            Write('Hàng đợi đã rỗng');
            Return;
          End;
    2) {Lưu lại phần tử loại bỏ}
        Y:=Q[F]
```

Thủ tục loại bỏ phần tử khỏi hàng đợi

```
Function CQDELETE(Q,F,R)
    3) {Thay đổi chỉ số}
        If F=R then F:=R:=0
        Else If F=n then F:=1
            Else  F:=F+1;
    4) CQDELETE:=Y;
Return
```