Mục lục:

ĐỀ CƯƠNG CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

➔ *Khi tìm hiểu 1 CTDL, cần phải tìm hiểu những nội dung gì? Vì sao?*

- Đặc điểm tổ chức:

  CTDL là tập hợp các DL nguyên tử, các DL này đc liên kết vs nhau = 1 cách nào đó

- Cấu trúc lưu trữ:

  CTLT là cách biểu diễn 1 CTDL trong bộ nhớ ➔ lưu trữ ntn?

- Các phép toán trên cấu trúc:

  GT gắn liền vs CT. khi tìm hiểu các phép toán thì ta tìm hiểu dưới dạng giả mã, gắn vs 1 CTLT cụ thể.

  Vì sao phải tìm hiểu phép toán: CTDL nào thì GT đó. Nên khi tìm hiểu CTDL nào thì phải căn cứ vào GT cho phép toán đó

Chương 2: Mảng và Danh sách

- ✓ MẢNG.
- ✓ DANH SÁCH
- ✓ CẤU TRÚC NGĂN XẾP
- ✓ CẤU TRÚC HÀNG ĐỢI

1. MẢNG.

a. Đặc điểm tổ chức.

- Mảng là một tập hợp có thứ tự gồm một số cố định các phần tử cùng kiểu.
- Các phần tử được truy nhập trực tiếp thông qua chỉ số.
- Mảng có thể tổ chức thành 1 chiều, 2 chiều,…,n chiều

  Vs mảng 1 chiều thì mỗi phần tử đc xđ = 1 chỉ số

  Mảng 2 chiều thì mỗi phần tử xđ = 2 chỉ số

  …

  Mảng n chiều thì mỗi phần tử đc xđ = n chỉ số

- Số lượng các phần tử cố định và xác định.
- Mảng được lưu trữ bằng cấu trúc kế tiếp và chỉ sử dụng cấu trúc lưu trữ này vì các phần tử của mảng đc *truy nhập trực tiếp*.
- Có phép tạo lập mảng, tìm kiếm, truy nhập một phần tử từ mảng. Không có phép bổ sung hoặc loại bỏ một phần tử của mảng.

b. Cấu trúc lưu trữ.

Sử dụng CTLT ***kế tiếp*** để lưu trữ các phần tử Dl của mảng

Đc lưu trữ trong các ô nhớ liền kề nhau

Dùng vecto lưu trữ V gồm n ô nhớ có chỉ số từ 1 -> n

a1 -> V[1]

a2 -> V[2]

…

an -> V[n]

V

a1	a2	a3	a4	a5	…				
1	2	3	4	5	…				

Vs cách lưu trữ này, để truy nhập phần tử i -> ta sử dụng V[i]

- Vs mảng nhiều chiều: 1 phần tử xđ = nhiều chỉ số

  Vấn đề là 1 ô nhớ chỉ lưu trữ đc 1 phần tử -> lưu trữ kế tiếp như thế nào.

  Vd: lấy mảng 2 chiều minh họa chùg cho mảng nhiều chiều. với mảng 2 chiều có **m hàng** và **n cột**

➔ Khi lưu trữ, ta lưu trữ lần lượt hết hàng 1 đến hàng 2, hết hàng 2 đến hàng 3,…

Xét m = 3, n = 4

A11	A12	A13	A14
A21	A22	A23	A24
A31	A32	A33	A34

➔ Ta có vecto V:

A11	A12	A13	A14	A21	A22	A23	A24	A31	A32	A33	A34
1	2	3	4	5	6	7	8	9	10	11	12

A_{ij} => V[k]

➔ k = f(i,j) = (i - 1)*n + j

c. Các phép toán.

- Có phép tạo lập mảng, tìm kiếm, truy nhập một phần tử từ mảng. Không có phép bổ sung hoặc loại bỏ một phần tử của mảng.

2. DANH SÁCH

a. Danh sách.

- 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.
- Danh sách có thể rỗng.
- Danh sách được lưu trữ bằng cả cấu trúc kế tiếp và cấu trúc phân tán.
- Các phép toán trên danh sách là:
  - Phép bổ sung: Có thể bổ sung phần tử vào danh sách.
  - Phép loại bỏ: có thể loại bỏ một phần 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.
- 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.

b. 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<= i <= 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 <=i
- n thì ai+1 gọi là phần tử sau ai ; 2 <=i <= n thì phần tử ai-1 là phần tử trước của ai .
- 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).
- 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
- 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 (kế tiếp!)

a. Đặc điểm tổ chức:

- Trình bày *danh sách là gì*? Là danh sách có thứ tự 1 số biến động cùng kiểu.
- Trình bày *danh sách tuyến tính là gì*? Là danh sách mà quan hệ giữa các phần tử đc xđ rõ rãng.
- *Ngăn xếp* 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).
- *Ngăn xếp có thể rỗng* hoặc bao gồm một số phần tử.

b. Cấu trúc lưu trữ:

- Có thể lưu trữ ngăn xếp bởi một vector S gồm n ô nhớ kế tiếp nhau có chỉ số từ 1 đến n.
- Dùng biến chỉ số T để lưu trữ chỉ số đỉnh ngăn xếp
  T thay đổi thì ngăn xếp hoạt động
- Ta quy ước khi ngăn xếp rỗng -> T=0.

Hoặc vẽ hình trong vở.

c. Các phép toán:

Ngăn xếp có 3 phép toán chính:

- Push: Bổ sung.
- Pop: Loại bỏ.

- IsEmpty: Kiểm tra rỗng.

1) 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.}

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

2) 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.}

```
Function  Pop(Var S,T)
    1)  {Kiểm tra ngăn xếp rỗng?}
        If  T = 0 then
                Begin
                    Write('Ngăn xếp rỗng!');
                    Return;
                End
    2) {Lưu lại phần tử đỉnh}
    Tg := S[T];
    3)  {Thay đổi chỉ số đỉnh}
    T := T-1;
    4)  {Trả về phần tử đỉnh}
        Pop := Tg;
Return
```

3)Kiểm tra ngăn xếp rỗng:

- Vào: ngăn xếp (S, T).

- Ra: trả về true nếu ngăn xếp rỗng. ngược lại trả về false nếu ngăn xếp không rỗng.

{Hàm này nhận vào ngăn xếp được lưu trữ bởi vecto S có n ô nhớ thông qua chỉ số. trả về giá trị True nếu ngăn xếp rỗng, nếu không thì trả về False nếu ngăn xếp không rỗng}

```
Function IsEmpty(S, T)
    If T=0 then IsEmpty:=True
    Else IsEmpty:=False;
Return
```

4. CẤU TRÚC HÀNG ĐỢI (kế tiếp!).

A. Đặc điểm tổ chức của ngăn xếp.

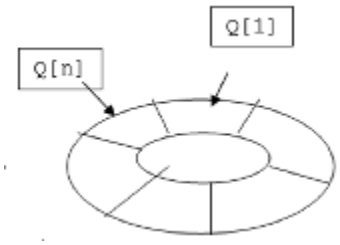
- Trình bày danh sách là gì?
- Trình bày danh sách tuyến tính là gì?
- Hàng đợi (queue) là kiểu danh sách tuyến tính mà phép bổ sung được thực hiện ở một đầu, gọi là lối sau (rear) và phép loại bỏ thực hiện ở một đầu khác, gọi là lối trước (front).
- Phép bổ sung và loại bỏ phần tử của hàng đợi được 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.

B. Cấu trúc lưu trữ:

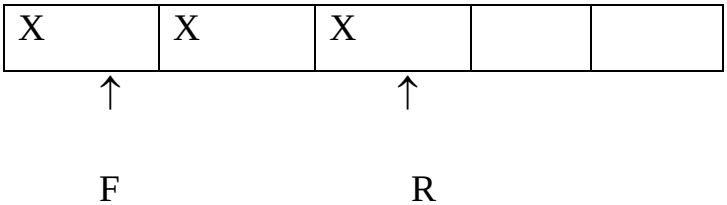
1. Lưu trữ bằng mảng (lưu trữ kế tiếp)

- 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 hàng đợi thì R sẽ tăng lên 1, còn khi loại bỏ một phần tử khỏi hàng đợi thì F tăng lên 1.

- Khi hàng đợi rỗng thì R=F=0.



Q



C. Các phép toán:

a. Bổ sung 1 phần tử vào hàng đợi

- Vào : phần tử dữ liệu x, hàng đợi (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(Var Q, F, R; x)

    1)  {kiểm tra hàng đợi đầy}
        If (F=1) and ( R=n) or (R+1=F) then
                Begin
                        Write('Hàng đợi đầy!');
                        Return;
                End;

    2){Thay đổi chỉ số R}
            If  F=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
```

b. Loại bỏ phần tử ra khỏi Hàng đợi

- 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}

```
Function CQDelete(Var Q,F,R)
    1) {Kiểm tra hàng đợi rỗng}
       If F=R=0 then
          Begin
              Write('Hàng đợi đã rỗng!');
              Return;
          End;
    2) {Lưu lại phần tử loại bỏ}
       Tg:=Q[F]
    3) {Thay đổi chỉ số F}
       If F=R then F:=R:=0
       Else if F=n then F:=1
       Else F:=F+1;
    4) {Trả về phần tử loại bỏ}
       CQDelete:=Tg;
```

Return

c. Kiểm tra hàng đợi rỗng:

- Vào: Q,F,R

- Ra: Trả về True nếu hàng đợi rỗng. ngược lại trả về False nếu hàng đợi không rỗng

{Hàm này nhận vào hàng đợi qua đối số và trả về giá trị True nếu hàng đợi rỗng, nếu không thì trả về False nếu hàng đợi không rỗng}

```
Function CQIsEmpty(Q,F,R)
    If F=R=0 then CQIsEmpty:=True
    Else CQIsEmpty:=False;
Return
```

d. Kiểm tra hàng đợi đầy:

- Vào: Q,F,R

- Ra: Trả về True khi hàng đợi đầy, nếu không thì trả về False khi hàng đợi không đầy

{Hàm này nhận vào hàng đợi qua đối số và trả về giá trị True nếu hàng đợi đầy, nếu không thì trả về False nếu hàng đợi không đầy}

```
Function CQIsFull(Q,F,R)
    If F=1 and R=n or R+1=F then
        CQIsFull:=True
    Else CQIsFull:=False;
```

Return

Câu hỏi ôn tập chương 2:

Câu 11: Khi tìm hiểu một cấu trúc dữ liệu ta cần tìm hiểu những nội dung nào? Tại sao?

Câu 12: Hãy trình bày cấu trúc lưu trữ của mảng.

Câu 13: Hãy trình bày đặc điểm tổ chức của cấu trúc ngăn xếp.

Câu 14: Hãy trình bày cấu trúc lưu trữ ngăn xếp bằng mảng.

Câu 15: Viết giả mã một trong các phép toán của cấu trúc dữ liệu ngăn xếp.

Câu 16: Hãy trình bày đặc điểm tổ chức của hàng đợi.

Câu 17: Trình bày cấu trúc lưu trữ hàng đợi bằng mảng.

Câu 18: Viết giả mã các phép toán của cấu trúc dữ liệu hàng đợi.