# CHƯƠNG 7: THUẬT TOÁN QUY HOẠCH ĐỘNG VÀ ÁP DỤNG

# Quy hoạch động

1. Các bài toán con chung lồng nhau và giải thuật quy hoạch động
2. Giải thuật quy hoạch động giải bài toán cái túi
3. Giải thuật quy hoạch động giải bài toán dãy con lớn nhất
4. Giải thuật quy hoạch động giải bài toán dãy con chung dài nhất.
5. Giải thuật quy hoạch động giải nhân dãy ma trận.

# Các bài toán con chung lồng nhau và giải thuật quy hoạch động

- Ví dụ về bài toán con chung lồng nhau
- Quy hoạch động là gì?
- Ba giai đoạn của bài toán quy hoạch động

# Các bài toán con chung lồng nhau trong giải thuật chia để trị

- Khi chia bài toán thành các bài toán con, trong nhiều trường hợp, các bài toán con khác nhau lại chứa các bài toán con hoàn toàn giống nhau. Ta nói rằng chúng chứa các bài toán con chung giống nhau

Ví dụ:

# Ví dụ về bài toán con lồng nhau
# Tính số Fibonaci thứ n

**Định nghĩa số Fibonaci F(n):**

- F(0)=0
- F(1)=1
- F(n)=F(n-2)+F(n-1) với n>1

Ví dụ:

F(2)=1, F(3)= 2, F(4) = 3 , F(5)=5, F(6)=8

# Ví dụ: Tính số Fibonaci thứ n

**Tính theo đệ quy {top down}:**

**Function R_Fibonaci(n);**

- If n<2 then return n
- else
- R_Fibonaci(n):=R_Fibonaci(n-1)+R_Fibonaci(n-2);

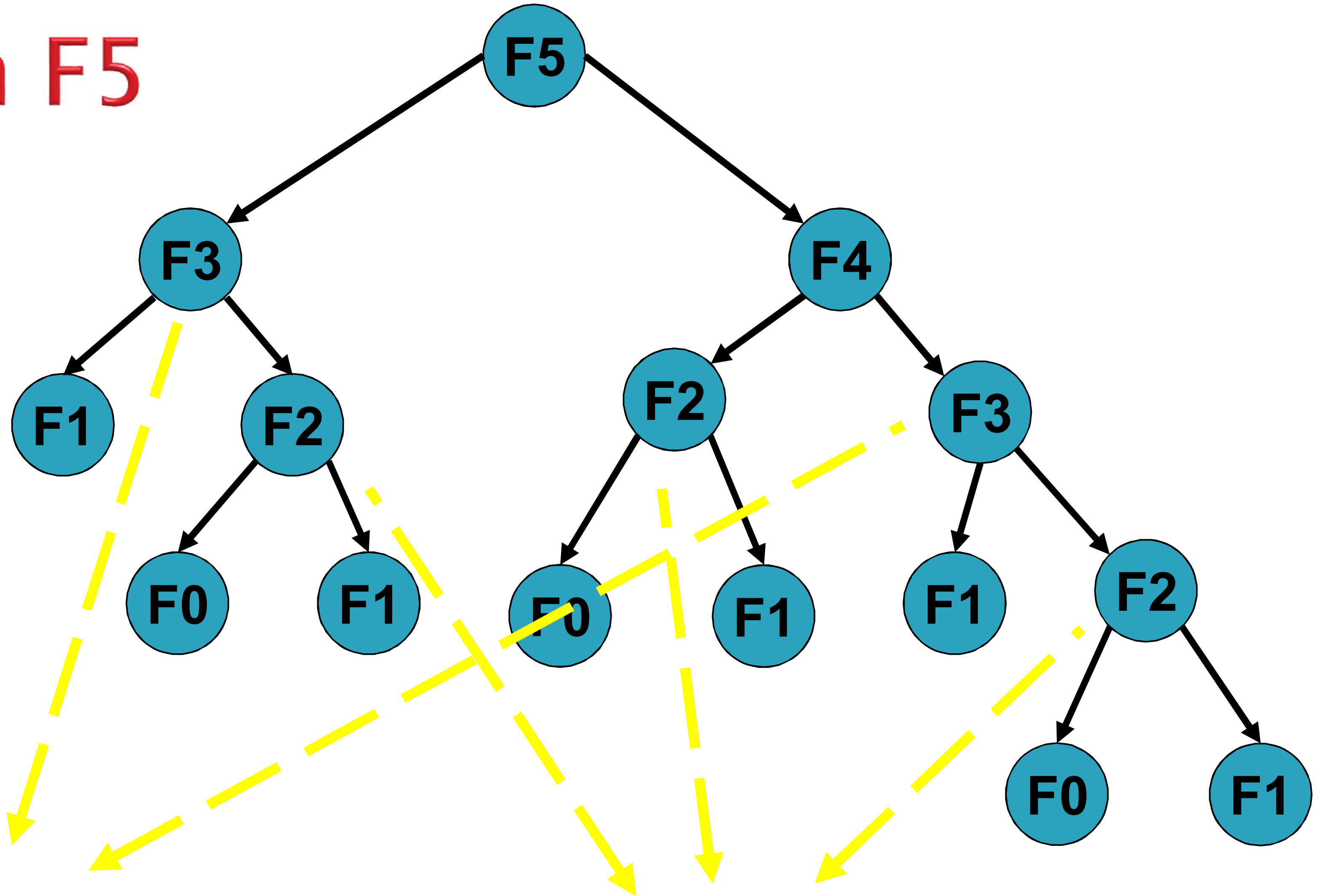
# So sánh hai giải thuật

- Khi tính F(5):
- Giải thuật đệ quy tính
- F(5) = F(3)+F(4)
- Tính F(3) F(3)= F(2)+F(1)
- F(2)=F(1)+F(0) = 1
- F(3)= 1+1= 2
- Tính F(4) F(4)= F(2)+F(3)
- F(2)= F(0)+F(1) = 1
- F(3)=F(1)+F(2) = 1+F(2)
- F(2)= F(0)+F(1) = 2
- F(3)= 1+2 =3
- F(4) = 2+3 = 5
- Tổng hợp F(5) = 3+5 =8

**Để tính F(5):**

- 2 lần tính F(3)
- 3 lần tính F(2)

# Tính F5




- 2 lần tính F(3)
- 3 lần tính F(2)

# Dùng Quy hoạch động để tính số Fibonaci thứ n

**Function Fibonaci(n);**

```
▪ If n < 2 then f:= n
▪ else
▪ begin f_0:=0 ; f_1:= 1;
▪ For k:=2 to n do
▪     begin
▪         f:=f_0+f_1 ; f_0:= f_1; f_1:= f;
▪     end;
▪ end;
▪ Return f;
```

# Quy hoạch động là gì?

**Quy hoạch động là một kỹ thuật thiết kế thuật toán trong đó:**

- Bài toán được chia thành những bài toán con kích thước nhỏ hơn và giải chúng một cách độc lập, ghi lại các kết quả, để tổng hợp thành lời giải của bài toán ban đầu

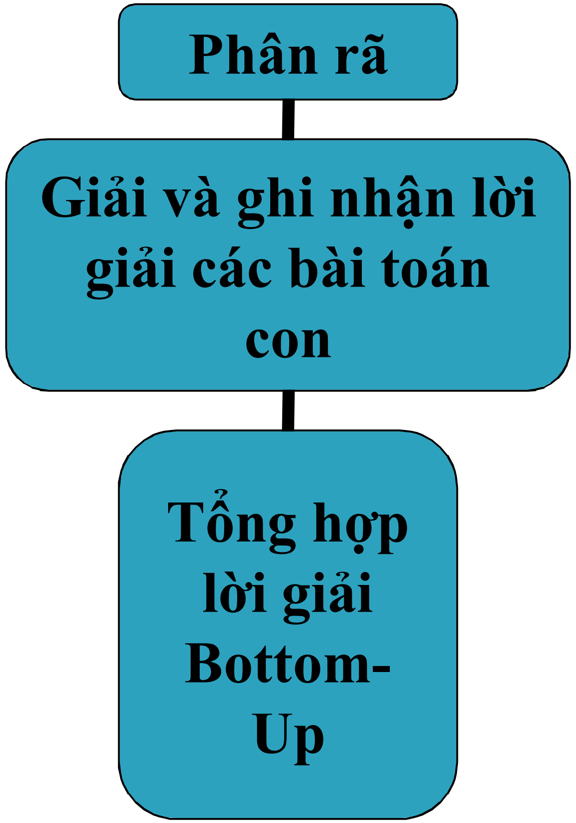
- **Khác với chia để trị:**
  - **Trong giải thuật chia để trị:**
    - Các bài toán con độc lập, sau đó các bài toán con này được giải một cách đệ quy.
  - **Trong giải thuật quy hoạch động:**
    - Các bài toán con là không độc lập với nhau, nghĩa là các bài toán con cùng có chung các bài toán con nhỏ hơn.

# Ba giai đoạn của quy hoạch động

- Phân rã:
  - Chia bài toán cần giải thành những bài toán con nhỏ hơn có cùng dạng với bài toán ban đầu sao cho bài toán con kích thước nhỏ nhất có thể giải một cách trực tiếp. Bài toán xuất phát có thể coi là bài toán con có kích thước lớn nhất
- Giải các bài toán con và ghi nhận lời giải:
  - Lưu trữ lời giải của các bài toán con vào một bảng để sử dụng lại nhiều lần do đó không phải giải lặp lại cùng một bài toán.
- Tổng hợp lời giải:
  - Lần lượt từ lời giải của các bài toán con kích thước nhỏ hơn xây dựng lời giải của bài toán kích thước lớn hơn, cho đến khi thu được lời giải của bài toán xuất phát (là bài toán con có kích thước lớn nhất).

# Lược đồ quy hoạch động

- Kỹ thuật giải các bài toán con của quy hoạch động là quá trình đi từ dưới lên (bottom – up) là điểm khác quan trọng với phương pháp chia để trị, trong đó các bài toán con được trị một cách đệ quy (top – down).



# Các yếu tố của một giải thuật quy hoạch động giải bài toán tối ưu

- **Cơ sở của quy hoạch động**:
  - Những trường hợp đơn giản có thể tính trực tiếp
- **Cấu trúc con tối ưu:**
  - Phương pháp chia nhỏ các bài toán cho đến khi gặp được bài toán cơ sở.
- **Tổng hợp**:
  - Hệ thức truy hồi tính giá trị tối ưu của hàm mục tiêu của bài toán lớn qua giá trị tối ưu của các bài toán con thành phần.

# Hiệu quả của quy hoạch động

- Khi có các bài toán con lồng nhau, phương pháp chia để trị sẽ tỏ ra không hiệu quả, khi nó phải lặp đi lặp lại việc giải các bài toán con chung đó.
- Quy hoạch động sẽ **giải mỗi bài toán con một lần và lời giải của các bài toán con sẽ được ghi nhận**, để thoát khỏi việc giải lại bài toán con mỗi khi ta đòi hỏi lời giải của nó.
- Quy hoạch động thường được áp dụng để giải các bài toán tối ưu. Trong các bài toán tối ưu, ta có một tập các lời giải, và một hàm mục tiêu nhận giá trị số. Ta cần tìm một lời giải để hàm mục tiêu đạt giá trị nhỏ nhất hoặc lớn nhất.

# Các ví dụ áp dụng quy hoạch động

1. Bài toán Cái túi dạng 0-1
2. Bài toán dãy con chung dài nhất
3. Bài toán nhân dãy ma trận

**.... và nhiều bài toán khác**

# Bài toán cái túi (dạng 0-1).

**Bài toán**

- Một tên trộm tìm thấy n gói đồ vật, gói thứ i có khối lượng là w[i], có giá trị là v[i] (w[i],v[i]$\in$N), nhưng cái túi của anh ta chỉ có thể mang được khối lượng tối đa là **M** (M$\in$N). Vậy tên trộm chọn mang những gói nào?
- Trong bài toán cái túi dạng 0–1 tên trộm với mỗi gói đồ vật chỉ có thể lấy nguyên vẹn từng gói hoặc không lấy.

# Phân rã

**Giảm kích thước:**

Với các giá trị *i* và *L*: i = 1,2,.., n và L =0, 1, 2,..., M. Gọi **MaxV**(i,L) là tổng giá trị lớn nhất có thể chọn trong i đồ vật (1,.., i) với trọng lượng tối đa L.

**Bài toán con:**

**Trong dãy *i* đồ vật 1,.., i có thể**

- **Bài toán con 1**: Nếu có chọn vật thứ i (nếu w[i] ≤ L), khi đó giá trị lớn nhất có thể là: **MaxV(i–1, L– w[i]) + v[i]** ;
- **Bài toán con 2**: Nếu không chọn vật thứ i, khi đó giá trị lớn nhất là : **MaxV(i–1, L)**

**Tổng hợp**

**MaxV(i, L) =**

**max{MaxV(i –1,L– w[i]) +v[i] , MaxV(i –1,L)}**

# Công thức truy hồi

- **Trường hợp cơ sở**
- Nếu L = 0 thì MaxV(i,L) = 0 với mọi i=1,..,n

# Mã: Giải thuật Bag_Best

**Procedure Bag_best**

- {Khởi tạo}: For L: = 0 to  M do MaxV[0,L] :=0 ;
- For i = 1 to n do
-   For L = 1 to M do
-     Begin
-      MaxV[i,L] := MaxV[ i–1,L];
-      If  (w[i] ≤ L) and
        (MaxV[i–1,L–w[i]] + v[i] > MaxV[i-1, L])
         then   MaxV[i, L] := MaxV[i–1,L–w[i]]+v[i] ;
- End;
-  Return MaxV(n, M)

# Ví dụ

Có 6 đồ vật và tổng trọng lượng tối đa có thể mang là 10

W: 10

| j | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Khối lượng W(j) | 6 | 3 | 3 | 7 | 4 | 3 |
| Giá trị V(j) | 12 | 1 | 8 | 1 | 10 | 3 |

# Giải

| W | 10 | j | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| Khối lượng W(j) | | | 6 | 3 | 3 | 7 | 4 | 3 |
| Giá trị V(j) | | | 12 | 1 | 8 | 1 | 10 | 3 |

| i | w | v | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 6 | 12 | L-w(i) | - | - | - | - | - | 0 | 1 | 2 | 3 | 4 |
| | | | Yes | 0 | 0 | 0 | 0 | 0 | 12 | 12 | 12 | 12 | 12 |
| | | | Max | 0 | 0 | 0 | 0 | 0 | 12 | 12 | 12 | 12 | 12 |
| 2 | 3 | 1 | L'=L-w(i) | - | - | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | | | Max(i-1,L') | - | - | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 12 |
| | | | Yes | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 13 | 13 |
| | | | Max | 0 | 0 | 1 | 1 | 1 | 12 | 12 | 12 | 13 | 13 |
| 3 | 3 | 8 | L-w(i) | - | - | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | | | Max(i-1,L') | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 13 | 13 |
| | | | Yes | 8 | 8 | 8 | 8 | 9 | 9 | 9 | 9 | 20 | 20 |
| | | | Max | 8 | 8 | 8 | 8 | 9 | 12 | 12 | 12 | 20 | 20 |

# Bài toán dãy con chung dài nhất

- **Bài toán;**
- Cho hai dãy $X = (x_1,x_2,\ldots,x_m)$ và $Y = (y_1,y_2,\ldots,y_n)$. Cần tìm dãy con chung dài nhất của hai dãy X và Y.

# Phân rã .

Với mỗi $0 \le í \le m$ và $0 \le j \le n$ xét bài toán con :

- Tính C[i, j] là độ dài của dãy con chung dài nhất của hai dãy.
- $\mathbf{X_i} = x_1 x_2 \ldots x_i$ và $\mathbf{Y_j} = y_1 y_2 \ldots y_i$ . Chú y rằng ( $X_o$ và $Y_o$ là xâu rỗng)
- Như vậy ta đã phân bài toán cần giải ra thành $(m+1)\times(n+1)$ bài toán con. Bản thân bài toán xuất phát là bài toán con có kích thước lớn nhất C(m,n).

# Bài toán con cơ sở và tổng hợp

**Các bài toán con cơ sở**

- $C[0, j] = 0\ \forall j = 0..\ n$ và $C[i,0] = 0, i = 0..\ m$.
  (là độ dài dãy con chung lớn nhất của dãy rỗng với một dãy khác).

**TỔNG HỢP**

Với $i > 0, j > 0$ . Tính $C[i, j]$.

Có hai tình huống:

- Nếu $x_i = y_j$ thì dãy con chung dài nhất của $X_i$ và $Y_j$ sẽ thu được bằng việc bổ sung $x_i$ vào dãy con chung dài nhất của hai dãy $X_{i-1}$ và $Y_{j-1}$
- Nếu $x_i \neq y_j$ thì dãy con chung dài nhất của $X_i$ và $Y_j$ sẽ là dãy con dài hơn trong hai dãy con chung dài nhất của ($X_{i-1}$ và $Y_j$) và của ($X_i$ và $Y_{j-1}$) .

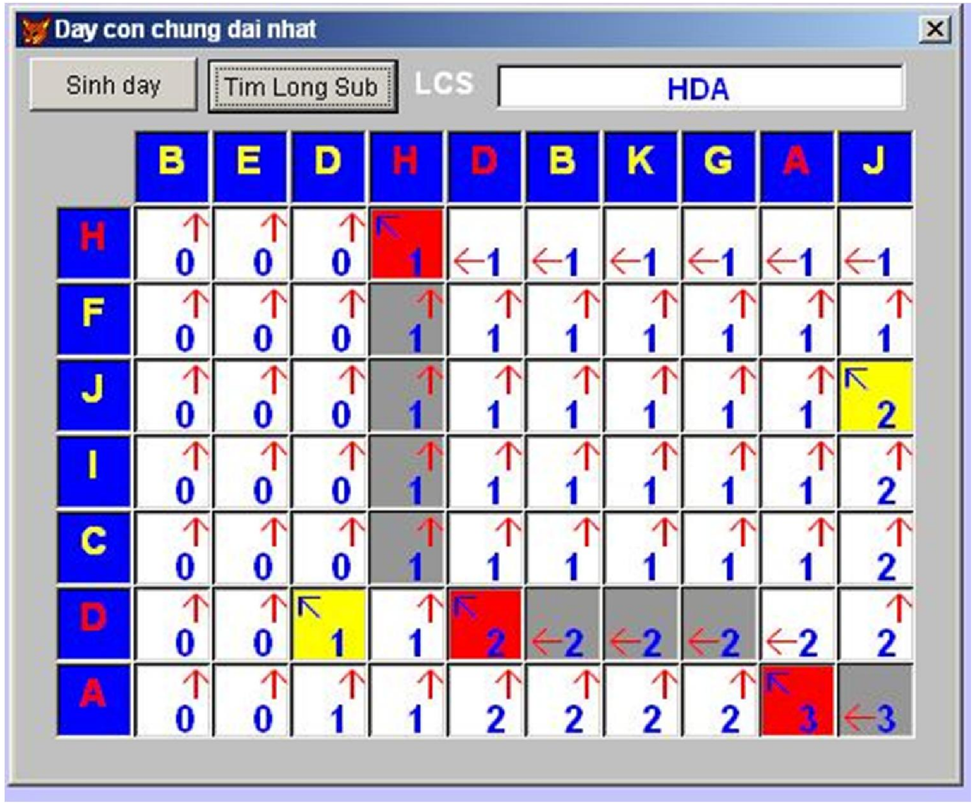
# Công thức truy hồi để tính C[i,j].

- C[i,j] = 0 nếu i =0 hoặc j=0
- C[i,j] = C[i-1,j-1]+1 nếu $x_i = y_j$
- C[i,j] = Max{ C[i-1,j], C[i,j-1]} nếu $x_i \neq y_j$

# Procedure LCS(X,Y)

- Begin
- {Khởi tạo}
-  For i :=1 to m do c[i,0]:=0;
-              For j: =1 to n do c[0,j ]:=0;
- {Tính từ dưới lên}
- For i: =1 to m do
-   for j: = 1 to n do
-    If $x_i = y_j$ then
-       begin c[i,j]:=c[i-1,j-1]+1; b[i,j]:='↖' end
-    else
-       If c [i-1,j]≥ c[i,j-1] then
-         begin c[i,j]:=c[i-1,j]; b[i,j]:='↑'; end
-       else
-         begin c[i,j]:=c[i,j-1]; b[i,j]:='←';end;
- End;

# Ví dụ: Dãy con chung dài nhất là HDA




| | B | E | D | H | D | B | K | G | A | J |
|---|---|---|---|---|---|---|---|---|---|---|
| H | ↑0 | ↑0 | ↑0 | ↖1 | ←1 | ←1 | ←1 | ←1 | ←1 | ←1 |
| F | ↑0 | ↑0 | ↑0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 |
| J | ↑0 | ↑0 | ↑0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↖2 |
| I | ↑0 | ↑0 | ↑0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑2 |
| C | ↑0 | ↑0 | ↑0 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑1 | ↑2 |
| D | ↑0 | ↑0 | ↖1 | ↑1 | ↖2 | ←2 | ←2 | ←2 | ←2 | ↑2 |
| A | ↑0 | ↑0 | ↑1 | ↑1 | ↑2 | ↑2 | ↑2 | ↑2 | ↖3 | ←3 |

# Ví dụ

| | | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| | | **D** | **I** | **N** | **H** | **V** | **U** |
| 1 | **N** | ↑0 | ↑0 | ↖1 | ←1 | ←1 | ←1 |
| 2 | **I** | ↑0 | ↖ 1 | ←1 | ←1 | ←1 | ←1 |
| 3 | **N** | ↑0 | ↑1 | ↖ 2 | ← 2 | ← 2 | ← 2 |
| 4 | **H** | ↑ 0 | ↑ 1 | ↑ 2 | ↖ 3 | ← 3 | ← 3 |
| 5 | **C** | ↑ 0 | ↑ 1 | ↑ 2 | –3 | ↑ 3 | ↑ 3 |
| 6 | **U** | ↑ 0 | ↑ 1 | ↑ 2 | ↑ 3 | ↑ 3 | ↖ 4 |

- **Nếu X[ i ]=Y[ j ] thì lấy giá trị ô đứng hàng trên bên trái + 1**
- **Nếu X[ i ] ≠ Y[ j ] thì lấy theo giá trị lớn hơn trong hai giá trị đứng trên hoặc đứng trước**

# Bài toán dãy con liên tiếp có tổng lớn nhất

- Cho dãy  A dưới dạng mảng A[1..n ] các số
- Hãy tìm dãy con các phần tử liên tiếp của dãy A có tổng lớn nhất

- Ví dụ:

# Phân rã

- **Gọi S(i) là tổng của dãy con lớn nhất trong dãy i phần tử**

  $A_i$ = a[1], …., a [i], i = 1,2,…, n thì S(n) là giá trị cần tìm.

- **Bài toán con cơ sở**

  Với i =1 ta có S(1)= a[1].

## Phân rã

Giả sử i > 1 và S[k] là đã biết với k = 1,.., i–1.

- Ta cần tính S[i] là tổng của dãy con liên tiếp lớn nhất của dãy a[1]…, a[i-1], a[i].

Các dãy con liên tiếp của dãy này có thể là một trong hai trường hợp:

- Các dãy con liên tiếp có chứa a[i]
- Các dãy con liên tiếp không chứa a[i]

Gọi MaxS(i) là tổng lớn nhất của các dãy con liên tiếp của dãy a[1]...a[i]

MaxE(i) là tổng lớn nhất của các dãy con liên tiếp của dãy a[1]..a[i] chứa chính a[i].

# Phân rã .....

- Tổng lớn nhất của các dãy con liên tiếp của dãy a[1]..a;[i] không chứa a[i] chính là tổng lớn nhất của các dãy con của dãy a[1]..a[i-1]$_1$, nghiã là MaxS(i–1).

Do đó

MaxS(i) = max { MaxS(i–1) , MaxE(i)}.

# Tính MaxE(i)

- Để tính MaxE(i), i = 1, 2, …, n, ta cũng có thể sử dụng công thức đệ quy như sau
- 1. Với i=1: MaxE(i) = a[1];
- 2.Với i >1, Gọi S là dãy con kế tiếp lớn nhất của dãy a[1]..a[i] có chứa a[i]. Có hai khả năng:
- Nếu S chứa  a[i–1] do đó độ dài lớn nhất có thể là MaxE(i–1)+a[i];
- Nếu S không chứa a[i–1]  thì S chỉ gồm a[i]
- Do đó: *MaxE[i] = max {a[i] , MaxE[i–1] + a[i] }, i > 1.*

# Procedure Maxsub(a);

*Ý nghĩa các biến:*

- *maxS: tổng dãy con lớn nhất*
- *maxE: tổng dãy con có chứa phần tử cuối lớn nhất*
- *s,e chỉ số đầu và cuối của dãy con có tổng lớn nhất*
- *s1 chỉ số đầu của dãy lớn nhất kết thúc tại i*

```
Var MaxS,MaxE, s, e, e1 :Integer ;
Begin
 MaxS:=a[1]; MaxE:= a[1]; s:=1; e:=1; s1:=1;
For i: = 2 to n do
   begin
        if MaxE>0 then MaxE:=MaxE+a[i]
         else begin MaxE = a[i]; s1:=i;end;
       if (MaxE > MaxS) then
         begin MaxS:= MaxE; e:=i;
               s:=s1;end;
   End;
End;
```

# Ví dụ Dãy con có tổng lớn nhất

| i | a[i] | MaxE | s1 | MaxS | s | e | Dãy con tổng lớn nhất |
|---|---|---|---|---|---|---|---|
| 1 | 13 | 13 | 1 | 13 | 1 | 1 | 13, |
| 2 | -15 | -2 | 1 | 13 | 1 | 1 | 13, |
| 3 | 2 | 2 | 3 | 13 | 1 | 1 | 13, |
| 4 | 18 | 20 | 3 | 20 | 3 | 4 | 2, 18, |
| 5 | 4 | 24 | 3 | 24 | 3 | 5 | 2, 18, 4, |
| 6 | 8 | 32 | 3 | 32 | 3 | 6 | 2, 18, 4, 8, |
| 7 | 0 | 32 | 3 | 32 | 3 | 6 | 2, 18, 4, 8, |
| 8 | -5 | 27 | 3 | 32 | 3 | 6 | 2, 18, 4, 8, |
| 9 | -8 | 19 | 3 | 32 | 3 | 6 | 2, 18, 4, 8, |

# Nhân dãy ma trận

Bài toán: Khi nhân hai ma trận $A_{mn}$ và $B_{n,p}$ ta dùng ba vòng For

```
For i: = 1 to m do
    For j := 1 to p do
    Begin C[i,j] := 0;
      For k:=1 to n do
          C[i,j]:=   C[i,j]+a[i,k]*b[k,j];
    End;
```

*Số các phép nhân phải thực hiện là m*n*p.*

# Nhân dãy ma trận

- Xét phép nhân 3 ma trận $A_{3,4}$ x $B_{4,5}$ x $C_{5,6}$.

Có hai cách nhân

ABC=(AB)C và A(BC).

- Tính tích AB cần 3*4*5= 60 phép nhân được ma trận D cấp 3x5. Tính DC cần 3x5x6 = 180 phép nhân. Do đó tính (AB)C cần 60+180 = 240 phép nhân
- Tính tích (BC) cần 4*5*6= 120 phép nhân được ma trận E cấp 4x6; tính AE cần 3x4x6=72 phép nhân. Do đó tính A(BC) cần 120+72= 192 phép nhân.

# Bài toán nhân dãy ma trận

Xét phép nhân dãy ma trận

$$M_1M_2..M_n$$

1). Có bao nhiêu cách tổ chức thứ tự thực hiện phép nhân dãy ma trận này?

2). Nhân theo thứ tự nào để số phép nhân các số là ít nhất?

# Số cách thực hiện dãy phép nhân n ma trận

Ký hiệu T (n) là số cách điền các dấu ngoặc vào biểu thức tích của n ma trận. Giả sử ta định đặt dấu ngoặc phân tách đầu tiên vào giữa ma trận thứ i và ma trận thứ (i + 1) trong biểu thức tích, tức là:

$$M = (M_1 M_2 \dots M_i)(M_{i+1} M_{i+2} \dots M_n)$$

Khi đó có T(i) cách đặt dấu ngoặc cho thừa số thứ nhất (M1 M2 … Mi) và T(n - i) cách đặt dấu ngoặc cho thừa số thứ hai (Mi+1 Mi+2 … Mn) và từ đó T(i)T(n-i) cách tính biểu thức $(M_1 M_2 \dots M_i)(M_{i+1} M_{i+2} \dots M_n)$.

# Có bao nhiêu cách tính $M_1M_2...M_n$?

- Công thức truy hồi

$$\begin{cases} T(1) = 1 \\ T(n) = \sum_{i=1}^{n-1} T(i)T(n-i) \end{cases}$$

- Công thức hiện

$$T(n) = \frac{1}{n} C_{2n-2}^{n-1},$$

# Có bao nhiêu cách?

- Một số giá trị của T(n)

| *n* | *1* | *2* | *3* | *4* | *5* | *10* | *15* |
|---|---|---|---|---|---|---|---|
| **T(n)** | **1** | **1** | **2** | **5** | **14** | **4862** | **2674440** |

# Cách tính tối ưu?

- Cách nào đòi hỏi số phép nhân các số ít nhất

# Phân rã (Xác định cấu trúc con tối ưu).

- **Giả sử cách tính tối ưu tích của n ma trận đòi hỏi dặt dấu ngoặc tách đầu tiên giữa ma trận thứ i và thứ (i+1) của biểu thức tích, thì khi đó cả hai tích con ($M_1$ $M_2$ … $M_i$) và ($M_{i+1}$ $M_{i+2}$ … $M_n$) cũng phải được tính một cách tối ưu.**

- **Do đó đó số phép nhân cần phải thực hiện để nhân dãy ma trận là tổng:**
  ***số phép nhân cần thực hiện để nhân hai dãy con + số phép nhân cần thực hiện để nhân hai ma trận kết quả***

# Phân rã bài toán

Gọi $m_{ij}$ là số phép nhân ít nhất cần thực hiện để tính tích ($i \leq j$)

$$(M_i M_{i+1} M_{i+2} \dots M_j), \quad 1 \leq i \leq j \leq n$$

Giả sử kích thước của các ma trận được cho bởi véc tơ *d[0 … n]*, trong đó ma trận $M_i$ có kích thước $d_{i-1} \times d_i$, $i = 1, 2, 3, \dots n$.

# Trường hợp cơ sở

- Khi i = j thì $m_{ii} = 0$
- Giả sử $j = i+s$ với $s \geq 1$ và phép nhân cuối cùng tách từ vị trí thứ k
- $(M_i M_{i+1} \ldots M_k)(M_{k+1} \ldots M_{i+s-1} M_{i+s})$.
- tích thứ nhất là ma trận kích thước (i-1), k, tích thứ hai co kích thước k, i+s
- số các phép nhân ít nhất để tính tích theo công thức này là

$$m_{ik} + m_{k+1,i+s} + d_{i-1} d_k d_{i+s}$$

# Công thức đệ quy

- $1 < s < n$:
  $m_{i,\, i+s} = \min\{m_{ik} + m_{k+1,i+s} + d_{i-1}d_kd_{i+s} \mid i \le k < i+s\}$,
  $i = 1, 2, \ldots, n - s$.

# Ví dụ

- Tìm cách tính tối ưu cho tích của bốn ma trận $M_1M_2M_3M_4$ với các kích thước
  d = (2, 5, 4, 3, 7).

Ta có với s=1

| | | | |
|---|---|---|---|
| $m_{12}$ | $M_1M_2$ | $2 \times 5 \times 4$ | = 40 |
| $m_{23}$ | $M_2M_3$ | $5 \times 4 \times 3$ | = 60 |
| $m_{34}$ | $M_3M_4$ | $4 \times 3 \times 7$ | =84 |

Với s = 2, d = (2, 5, 4, 3, 7).

| | |
|---|---|
| $m_{12}$ | = 40 |
| $m_{23}$ | = 60 |
| $m_{34}$ | =84 |

Cần tính $m_{13}$, $m_{24}$

| | | | |
|---|---|---|---|
| $m_{13}$ | $M_1M_2M_3$ | $(M_1M_2\ )(M_3)$ | 64 |
| k=1 | $(M_1)(M_2\ M_3)$ | $m_{11} + m_{23} + d_0 \times d_1 \times d_3 =$ 0+60+2*5*3 | 90 |
| k=2 | $(M_1M_2)\ (M_3)$ | $m_{12} + m_{33} + d_0 \times d_2 \times d_3$ $=40+0 + 2 \times 4 \times 3$ | 64 |

Ví dụ với d = (2, 5, 4, 3, 7).

| | |
|---|---|
| $m_{12}$ | = 40 |
| $m_{23}$ | = 60 |
| $m_{34}$ | =84 |

- Tính $m_{24}$

| | | | |
|---|---|---|---|
| $m_{24}$ | $M_2M_3M_4$ | $(M_2M_3)\ (M_4)$ | 165 |
| k=1 | $(M_2)(M_3\ M_4)$ | $m_{22} + m_{34} + d_1{*}d_2{*}\ d_4 =$ 0+ 84+5*4*7 | 224 |
| k=2 | $(M_2M_3)\ (M_4)$ | $m_{23} + m_{44} + d_1 \times d_3 \times d_4$ $=60+0+5 \times 3 \times 7$ | 165 |

## Ví dụ với d = (2, 5, 4, 3, 7).

- Với s = 3, ta tính $m_{14}$ , k = 1, 2 , 3

| | | | |
|---|---|---|---|
| $m_{13}$ | **64** | $m_{12}$ | **40** |
| $m_{24}$ | **165** | $m_{23}$ | **60** |
| $m_{34}$ | **84** | | |

| $m_{14}$ | $M_1M_2M_3M_4$ | $(M_1M_2M_3) M_4$ | |
|---|---|---|---|
| k=1 | $M_1(M_2M_3 M_4)$ | $m_{11} + m_{24} + d_0 * d_1 * d_4$<br>0+165+2*5*7 | 235 |
| k=2 | $(M_1M_2) (M_3 M_4)$ | $m_{12} + m_{34} + d_0d_2d_4 =$<br>40+84+2*4*7 | 180 |
| k=3 | $(M_1M_2M_3) M_4$ | $m_{13} + m_{44} + d_0d_3d_4$<br>64+0+2*3*7 | 106 |

Ví dụ với d = (2, 5, 4, 3, 7).

- **Tổng hợp kết quả**
- Tính tối ưu $M_1M_2M_3\ M_4$ là tính $(M_1M_2M_3)\ M_4$ với 126 phép nhân các số
- Tính tối ưu $(M_1M_2\ M_3)$ là tính $(M_1)(M_2\ M_3)$

Trả lời:

Với dãy các kích thước đã cho cách tính tối ưu là **$(M_1(M_2\ M_3))M_4$.**

# Độ phức tạp tính toán

- Với mỗi s thỏa mãn 1 < s < n, ta tính :
  $m_{i,\, i+s} = \min \{m_{ik} + m_{k+1, i+s} + d_{i-1}d_kd_{i+s} \mid i \le k < i+s\}$,
  $i = 1, 2, \ldots, n - s$.

- Với mỗi s > 0, có n – s phần tử trên đường chéo cần tính, để tính mỗi phần tử đó ta cần so sánh s giá trị số tương ứng với các giá trị có thể của k. Từ đó suy ra số phép toán cần thực hiện theo thuật toán là cỡ

# Độ phức tạp tính toán

tương đương với

$$\sum_{s=1}^{n-1}(n-s)s = n\sum_{s=1}^{n-1}s - \sum_{s=1}^{n-1}s^2$$

$$= n^2(n-1)/2 - n(n-1)(2n-1)/6$$

$$= (n^3 - n)/6$$

$$= 0(n^3)$$

# Mã giả tựa Pascal tính $m_{ij}$

```
Begin
For i: = 1 to n do m[i,i]:=0;
For s:=1 to n do
  For i:= 1 to n–s do
    begin  j:=i+s–1; m[i,j]:= +∞;
       For k:=i to j–1 do
       begin   q:=m[i,k]+m[k+1,j]+d[i-1]*d[k]*d[j];
            If(q<m[i,j]) then
                 begin   m[i,j]= q;   h[i,j] = k;
                 end;
       end;
    end;
End;
```

# Nhân hai ma trận với mảng h[i,j] tính từ thủ tục trên

```
Procedure Mult(i,j);
Begin
        If(i<j) then
         Begin
                k := h[i,j];
                X := Mult(i,k);
                Y := Mult(k+1,j)
            Return X*Y; {Nhân ma trận X và Y}
          End
         Else
        Return M[i];
End;
```

# Bài tập

- Tìm cách nhân tối ưu để tính tích của dãy ma trận

$$A1 \times A2 \times A3 \times A4$$

trong đó vectơ kích thước của chúng là (2,4,5,3,2)