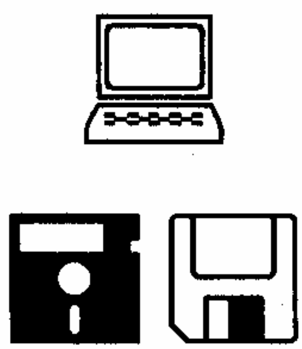
Trần Lê Hồng Dũ & Phạm Ngọc Chí Nhân
Trường PTTH Bến Tre

Các Bài Toán về

Quy Hoạch Động



Năm Học 1997-1998

Lời nói đầu

Có rất nhiều phương pháp để giải một bài toán Tin Học .Tôi xin giới thiệu vài nét về phương pháp **Quy Hoạch Động** và một số bài tập về phương pháp này để chúng ta cùng trao đổi và đút kết kinh nghiệm cho mình về một phương pháp để giải một bài tập khó.Xin đề ra một phương pháp xin các bạn cùng rút kinh nghiệm với chúng tôi.

Phương Pháp giải bài tập Tin Học:

Xét đưa bài toán về dạng mô hình quen thuộc :

* **Quy hoạch động:**

- Tìm công thức truy hồi (đệ quy) thường xuất phát từ trương hợp đơn giản có thể tìm ngay ra nghiệm.
- Tìm dữ liệu thích hợp .

* **Đồ thị:**

- Giải thuật tìm kiếm duyệt (sâu ,rộng)
- Chú ý tận dụng khai báo hằng.

* **Luồng cực đại trong mạng :**

- Các dạng toán có thể đưa về luồng.
- Các trọng số ban đầu.
- Điều kiện có nghiệm (tối ưu không ?)

* **Phương án bảng:**

* **Sơ đồ mạng lưới:**

* **Duyệt đệ quy:**

- Tổ chức quay lui.
- Thu hẹp không gian mẫu bởi các giá trị đề cử.
- Các hàm ước lượng Heuristic.

* **Các thuật giải đặt hiệu :**

- Về hoán vị,tổ hợp ,chỉnh hợp..v..v..

* **Các thuật toán hình học:**

- Trái phải
- Bao lồi.
- Phân chia đa giác
- Đường gấp khúc
- Trong ngoài đa giác

* **Xử lý BIT**

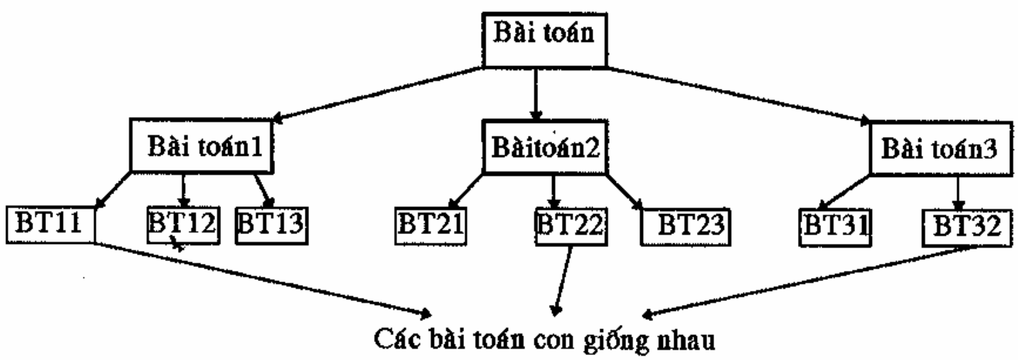
*** Các thuật toán xắp xếp:**(Quicksort,Heapsort,...)
*** Trò chơi:**các chiến thuật thắng
Chú ý : cần nhìn bài toán từ nhiều góc độ ,khía cạnh.
- Từ dể đến khó,đơn giản đến phức tạp
- Đôi khi không nên phức tạp hóa bài toán đơn giản

Vấn đề quan trọng cuối cùng khi vào thi: TỰ TIN,BÌNH TỈNH,CHÍNH XÁC,SÁNG TẠO !

Có phải mọi bài toán tối ưu đều có thể giải bằng phương pháp QHĐ không ?

* Nếu như bài toán tối ưu ta tìm được công thức truy hồi thì sẽ giải quyết được bằng QHĐ.

Bởi vì cơ sở của QHĐ:là nguyên lý chia để trị.Nó là một phương pháp cải tiến hơn của phương pháp giải bài toán theo hướng phân rả.Từ vấn đề lớn ta chia nó ra và tiếp tục như vậy cho đến khi gặp bài toán cở nhỏ có thể giải quyết dể dàng:

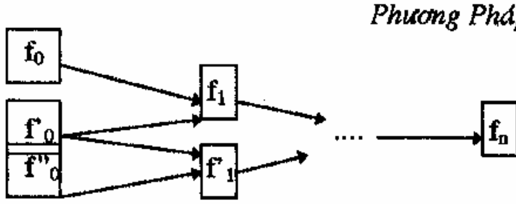


Phương pháp Phân Rã.

Nhưng khi giải theo hướng phân rã sẽ bị hạn chế về tốc độ chương trình do phải tính đi tính lại nhiều lần một số bài toán con giống nhau nào đó.Để khắc phục khuyết điểm này phương pháp **Quy Hoạch Động** đã ra đời kĩ thuật botton up , đi từ dưới lên.Đi từ trường hợp riêng đơn giản nhất có thể tìm ngay ra nghiệm.Kết hợp nghiệm các bài toán cở nhỏ ta thu được nghiệm bài toán cở lớn hơn và cứ tiếp tục như thế cho đến khi tìm được nghiệm của bài toán.

Trường hợp đơn giản:

Phương Pháp QHĐ



Hạn chế:

Tuy nhiên không phải lúc nào sự kết hợp lời giải của các bài toán con cũng cho ta lời giải của bài toán cở lớn hơn.

Số lượng bài toán con cần giải quyết và lưu trữ đáp án có thể rất lớn không thể chấp nhận được.vì dữ liệu và bộ nhớ máy tính không cho phép.

Tuy nhiên đa số các bài tóan tối ưu có thể đưa về phương pháp QHĐ để giải quyết một cách có hiệu quả.

Khi giải bài toán bằng phương pháp QHĐ ,phương trình truy hồi cần phải chính xác,cần phải chứng minh độ chính xác tin cậy của nó.

Bến Tre ngày 2 - 3 -1998

Các Bài Toán về Quy Hoạch Động

Bài toán 1.

Bài toán tìm xâu con chung dài nhất.

Xâu con chung được định nghĩa như sau: Nếu xóa đi một số kí tự của hai xâu thì hai xâu con còn lại của chúng bằng nhau.

<u>Ví dụ:</u> a=(CEACEEC)
b=(AECECA)

Kết quả dãy con chung dài nhất là:

c=(ECEC) hoặc c=(AEEC)

Có thể giải bài toán này theo nhiều cách.Nhưng bài toán này có thể sử dụng phương pháp Quy Hoạch Động với hiệu quả tốt hơn:

- Giả sử ta có dãy a có độ dài n,dãy b có độ dài m
- Giả sử : L(i,j) là độ dài lớn nhất của dãy con chung của 2 dãy:

$a_1 \ldots a_i$
$b_1 \ldots b_j$ (với $i \le n, j \le m$)

Ta thử đi tìm công thức đệ qui để tính L(i,j) .

Trường hợp đơn giản nhất . Nếu i=0 hoặc j=0 thì L(i,j)=0.

1./Nếu (i=0) or (j=0) L(i,j)=0

2./Nếu (i>0) and (j>0) and ai≠bj . ta có L(i,j)=Max(L(i-1,j),L(i,j-1))

3./Nếu (I>0) and (j>0) and ai=bj . ta có L(i,j)= 1 + L(i-1,j-1)

Phương Pháp:Dùng bảng để lưu kết quả của các bài toán con . mỗi lần cần đến ta chỉ cần truy xuất trong bảng.

Nếu ta viết một hàm tính độ dài theo hàm như sau:

```
Function L(i,j:byte):byte;
    Begin
        if (i=0)or(j=0) then L:=0
        else
            if a[i]=b[j] then
               L:=1+L(i-1,j-1)
            else
               L:=max(L(i-1,j),L(i,j-1));
    End;
```

Tuy đoạn chương trình trên vẫn cho kết quả đúng nhưng về thời gian bị chậm lại rất nhiều do tính lại nhiều lần kết quả bài toán con nào đó và đồng thời làm cho chương trình dễ bị tràn stack . Để khắc phục hạn chế đó ta Dùng mảng để lưu lại kết quả các bài toán con .Xuất phát từ trường hợp đơn giản nhất có thể tìm ra nghiệm.Kết hợp các nghiệm con đã có ta sẽ nhận được nghiệm của bài toán cở lớn hơn.Tiếp tục như thế cho tới khi tìm ra nghiệm của bài toán.

Ta chỉ cần duyệt qua một lần như sau để lập bảng:

```
For i:=1 to length(a) do
For j:=1 to length(b) do
   if a[i]=b[j] then
           L[i,j]:=1+L[i-1,j-1]
   else
           L[i,j]:=max(L[i-1,j],L[i,j-1]);
```

Khi đó L[n,m](hay L[length(a),length(b)]) sẽ cho kết quả là độ dài của xâu con chung dài nhất.

Để tìm xâu kết quả c:

Ta đi ngược từ ô L[n,m] hướng về ô L[0,0]

-Nếu ai=bj thì ta đặt ai (hoặc bj) vào bên trái dãy c(ở đầu xâu).

-Nếu $a_i \neq b_j$ thì ta tiến về ô L[i-1,j] trong trường hợp L[i-1,j]>L[i,j-1](ngược lại tức là L[i-i,j]≤ L[i,j-1] ta tiến về ô L[i,j-1]).

Với ví dụ trên ta có bảng sau:

i \ j	*0*	*1*	2	*3*	*4*	*5*	*6*
0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1
2	0	0	1	1	2	2	2
3	0	**1**	1	1	2	3	3
4	0	1	1	2	2	3	3
5	0	1	**2**	2	3	3	3
6	0	1	2	2	**3**	3	3
7	0	1	2	3	3	**4**	4

Hình 1.

Xem hình vẽ 1:mỗi lần đi theo → ta đã chọn được các ô khi $a_i=b_j$(các ô đậm)

⇒ Dãy kết quả là : c=(AEEC).

Qua thực hiện chương trình cho thấy dùng qui hoạch động đã khắc phục được nhiều khuyết điểm về thời gian.

Chương trình tìm xâu con chung dài nhất.

```
Program LongString;
      Uses Crt;
      Const
         Limit=250;
      Var  L:array[0..limit,0..limit] of byte;
         s1,s2,s3:string;
         f:text;
         m,n,i,j:byte;
{----------------------------}
Function max(var a,b:byte):byte;
       begin
       if a>b then max:=a
       else max:=b;
       end;
{----------------------------}
Begin
    clrscr;
    assign(f,'Xau.inp');
    reset(f);
    readln(f,s1);
    readln(f,s2);
    close(f);
    Fillchar(l,sizeof(l),0);
    m:=length(s1);
    n:=length(s2);
    for i:=1 to m do
       for j:=1 to n do
          if s1[i]=s2[j] then
            l[i,j]:=1+l[i-1,j-1]
          else
             l[i,j]:=max(l[i,j-1],l[i-1,j]);
    s3[0]:=chr(l[m,n]);
    i:=m;
    j:=n;
        repeat
        if s1[i]=s2[j] then
          begin
          insert(s1[i],s3,1)
```

```
            dec(i);
            dec(j);
          end
        else
          begin
            if l[i,j]=l[i-1,j] then dec(i)
            else dec(j);
          end
        until (i=0)or(j=0);
    clrscr;
    writeln(s3);
    write(l[m,n]);
End.
```

Bài toán 2. *Đề thi Tin Học Quốc Tế năm 1994 tại THỤY ĐIỂN*

7

3 5

8 1 0

2 7 4 4

4 5 2 6 5

Hình 2

Hình 2 biểu diễn tam giác số.Hãy viết chương trình tính tổng lớn nhất các số trên con đường bắt đầu từ đỉnh và kết thúc đâu đó ở đáy.

* Mỗi bước có thể đi chéo xuống phía trái hoặc đi chéo xuống phía phải.

* Số lượng hàng trong tam giác lớn hơn 1 nhưng ≤ 100

* Các số trong tam giác đều là số nguyên từ 0 đến 99.

Input data(dữ liệu vào)

Dữ liệu về số lượng của tam giác được đọc ra đầu tiên từ File INPUT.TXT

Ở ví dụ , file INPUT.TXT là như sau :

5
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5

Output data(dữ liệu ra)

Tổng lớn nhất được viết như là một số nguyên trong file ra:

OUTPUT.TXT

30

Bài này có nhiều thuật toán để giải.Có thể dùng đệ quy vét cạn hoặc dùng đánh giá nhánh cận để giải quyết, nhưng chắc chắn với n>10 chương trình sẽ mất khá nhiều thời gian và cả stack .

Có thể dùng QHĐ để giải bài toán này như sau :

Gọi M(i,j) là giá trị lớn nhất đi từ đỉnh tam giác tới ô (i,j) của tam giác.

Xét trường hợp đơn giản nhất :

- Với i=1 va j=1 thì M(i,j)=a[i,j]

(với a là mảng giá trị ban đầu của tam giác)

- Với j=1 hoặc j=i tức là ở bìa của tam giác (chỉ có một con đương duy nhất để đi đến). Khi đó :

M(i,j):=a[i,j]+M(i-1,j)   (nếu j=1)
M(i,j):=a[i,j]+M(i-1,j-1)   (nếu j=i)

- Với $j \neq 1$ và $j \neq i$ có thể đi tới (i,j) từ một trong hai ô sau (i-1,j-1) và (i-1,j) Khi đó :

M(i,j):=a[i,j]+max(M(i-1,j-1),M(i-1,j))

Ta sẽ có bảng sau khi lập.Và giá trị lớn nhất của đường đi là giá trị lớn nhất của dòng đáy của tam giác.

Val_{max} = max(M(n,j))(với j=1..n)

Với ví dụ trên ta có bảng M như sau:

7
10 15
18 16 15
20 **25** 20 19
24 **30** 27 26 24

Hình 3

Tuy đề bài không yêu cầu ta tìm ra đường đi cụ thể nhưng ta có thể tìm đường đi dựa vào bảng M một cách khá đơn giản.

Xuất phát từ ô kết quả (Ô có giá trị lớn nhất ở đáy tam giác M)

Tại ô (i,j) ta đi ngược lên chọn ô lớn trong hai ô kề tên nó (M(i-1,j-1)và M(i-1,j)

Chú ý khi tạo mảng M ta tạo M[0..limit,0..limit] khởi trị tất cả bằng 0 để khỏi phải xử lý riêng trường hợp (j=1 hoặc j=i).

Chương trình cụ thể:

```
Program BaiTamgiac;
    Uses Crt;
    Const
       Maxn=100;
    Var  a:array[0..maxn,0..maxn] of integer;
       n:byte;
       time:longint absolute 0:$46c;
       time1:longint;
{--------------------------------}
Procedure init;
      var i,j:byte;
         f:text;
     Begin
         assign(f,'Input.txt');
         reset(f);
         readln(f,n);
         fillchar(a,sizeof(a),0);
         for i:=1 to n do
         begin
            for j:=1 to i do
                read(f,a[i,j]);
            readln(f);
         end;
         close(f);
      end;
{----------------------------------}
Function Max(a,b:integer):integer;
      begin
      if a>b then max:=a
      else max:=b;
      end;
{----------------------------------}
Procedure Xuly;
```

```
        var i,j:byte;
          maxd:integer;
          f:text;
        Begin
          for i:=2 to n do
            for j:=1 to i do
              a[i,j]:=a[i,j]+max(a[i-1,j-1],a[i-1,j]);
          maxd:=0;
          For j:=1 to n do
            if maxd<a[n,j] then maxd:=a[n,j];
          assign(f,'Output.txt');
          rewrite(f);
          write(f,maxd);
          close(f);
        End;
BEGIN
   clrscr;
   time1:=time;
   init;
   xuly;
   write(time-time1);
END.
```

Bài toán 3. *Đề thi Tin Học Olympic 30-4-1997 tại Trường chuyên LÊ HỒNG PHONG (TP HCM)*

Một Khách cần sử dụng D[1],D[2],...,D[n] khăn trải bàn cho n ngày liên tiếp đánh số từ 1..n.Khách sạn có thể mua khăn trải bàn với giá A đồng 1 khăn ,hoặc thuê hiệu giặt trả nhanh (nhận lại khăn sạch vào ngay đầu ngày hôm sau) với giá B đồng 1 khăn, hoặc thuê hiệu giặt trả chậm(khăn dùng trong ngày i được giặt và trả lại vào đầu ngày i+2) với giá C đồng 1 khăn .(A>B>C) Giả sử trong ngày 1 khách sạn chưa có khăn.

Dữ liệu vào được cho trong file HOTEL.INP gồm 2 dòng.
dòng 1: Gồm 4 số nguyên dương n,A,B,C(n<100,A>B>C>0)
dòng 2: Gồm n số nguyên dương D[1],D[2],...,D[n]
các số trên cùng một dòng ghi cách nhau ít nhất một dấu cách.

Dữ liệu ra file HOTEL.OUT gồm n+1 dòng .
dòng 1 : ghi tổng chi phí nhỏ nhất.
dòng i+1 : ($1\le i \le n$) ghi 3 số nguyên không âm M[i] F[i] S[i] theo thứ tự là các số khăn cần mua,giặt trả nhanh ,giặt trả chậm trong ngày thứ i.

<u>Ví dụ:</u> HOTEL.INP

3 10 8 5
6 4 8
HOTEL.OUT
146
8 0 6
2 2 0
0 0 0

Có thể giải bài toán trên theo phương pháp QHĐ như sau.

Đầu tiên ta có thể bắt đầu bằng giả thiết tổng chi phí là lớn nhất.
Với ví dụ trên ta có bảng sau.

6 0 0
4 0 0
8 0 0

với chi phí là $180=F_{max}$

Ta duyệt từ dòng cuối cùng lên trên ở mỗi bước thứ I ta cố gắng tối ưu bớt chi phí $F_i \rightarrow min$.

Với ví dụ trên ta bắt đầu duyệt từ i=n=3

Tại dòng thứ i ta xét chiều như hình vẽ và cố gắng đẩy số d[i,1] lên các ô d[i-1,2] ,d[i-2,3] lớn nhất có thể (với d là mảng lưu các kết quả mua ,giặt trả nhanh ,trả chậm của khánh sạn trong n ngày)

Ta có : mảng d sau khi chuyển

6 0 6
4 2 0
0 0 0

với F=146.

Trường hợp đưa tổng quát đưa d[i,1]lên các ô khác với độ ưu tiên như sau :

+ Đầu tiên đưa lên ô d[i-2,3] số khăn tối đa có thể được (Số khăn chuyển không vượt quá số khăn mua trong ngày i-2 có nghĩa là d[i-1,3] ≤ d[i-2,1])cũng dễ hiểu vì chỉ mua có d[i-2,1] cái thì không thể giặt trả chậm số khăn vượt quá d[i-2,1].

Hay nói cách khác d[i,1] ≥ d[i,2]+d[i,3] (*) (∀i∈ 1..n)

+ Sau đó tại ô d[i,1] nếu còn dư khăn có nghĩa là d[i-2,1]-d[i-2,2] < d[i,1] vậy ta chỉ có thể chuyển tối đa d[i-2,1]-d[i-2,2] khăn số khăn còn lại ở ô d[i,1] lúc này là d[i,1] - (d[i-2,1]-d[i-2,2]) khăn.

+ Với qui tắc (*) ta tiếp tục chuyển số khăn còn lại ở ô d[i,1] lên ô d[i-1,2].

+ Nếu sau khi chuyển vẫn còn dư ta chuyển tất cả khăn còn lại từ ô d[i,1] lên ô d[i-1,1] có nghĩa là :

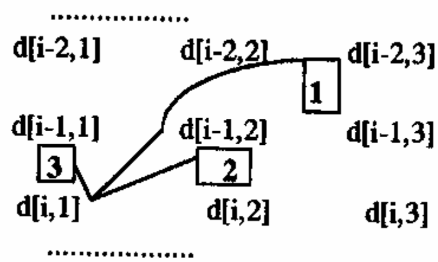
d[i-1,1]:=d[i-1,1]+d[i,1];

d[i,1]:=0;

sau đó ta giảm i xuống một đơn vị và tiếp tục chuyển d[i,1] lên các ô theo qui tắc như trên.

*** Chứng minh tính đúng đắn của giải thuật:**

- Ở mỗi bước ta tối ưu hóa từng phần theo sơ đồ sau với độ ưu tiên như hình vẽ:



Hình 4.

Rỏ ràng ở mỗi bước $F_i \le F_{i-1}$ do cách chuyển và giả thuyết (A<B<C)

Và ở mỗi bước tối ưu ta vẫn đảm bảo đủ số khăn sử dụng trong ngày thứ i do khăn giặt trả chậm trong ngày thứ i-2 và khăn giặt trả nhanh trong ngày i-1 sẽ được hiệu giặt sạch và trả lại trong ngày i.

Vậy sau quá trình chuyển $F_{max} \to \ldots \to F_i \to F_{i-1} \to \ldots \to F_{min}$.

Chương trình mẫu :

```
Program HoTel;
    Uses Crt;
    Const
      Maxn=100;
      fin='HOTEL.INP';
      fout='HOTEL.OUT';
    Var  a,b,c,n:byte;
      d:array[1..maxn,1..3] of byte;
      need:array[1..maxn] of byte;
{--------------------------------------}
Procedure Init;
     var f:text;
       i:byte;
     Begin
       assign(f,fin);
       reset(f);
       readln(f,n,a,b,c);
       for i:=1 to n do
           read(f,need[i]);
       close(f);
     End;
```

```
{---------------------------------------}
Procedure Chuyen(i1,j1,i2,j2:byte);
        var max:byte;
        Begin
            max:=d[i2,1]-d[i2,2]-d[i2,3];
            if max>=d[i1,j1] then
              begin
                  d[i2,j2]:=d[i1,j1];
                  d[i1,j1]:=0;
              end
            else
              begin
                  d[i2,j2]:=max;
                  d[i1,j1]:=d[i1,j1]-max;
              end;
        End;
{---------------------------------------}
Procedure Solve;
        var i:byte;
           max:byte;
        Begin
            fillchar(d,sizeof(d),0);
            for i:=1 to n do
               d[i,1]:=need[i];
            for i:=n downto 3 do
               begin
                   chuyen(i,1,i-2,3);
                   if d[i,1]>0 then chuyen(i,1,i-1,2);
                   if d[i,1]>0 then
                   begin
                       d[i-1,1]:=d[i-1,1]+d[i,1];
                       d[i,1]:=0;
                   end;
               end;
            chuyen(2,1,1,2);
        End;
{---------------------------------------}
Procedure Result;
        Var i,j:byte;
```

```
        s:word;
        f:text;
    Begin
        s:=0;
        for i:=1 to n do
            for j:=1 to 3 do
            case j of
            1:s:=s+d[i,j]*a;
            2:s:=s+d[i,j]*b;
            3:s:=s+d[i,j]*c;
            end;
        assign(f,fout);
        rewrite(f);
        writeln(f,s);
        for i:=1 to n do
        begin
            for j:=1 to 3 do
                write(f,d[i,j]:4);
            writeln(f);
        end;
        close(f);
    End;
{---------------------------------------}
BEGIN
    Init;
    Solve;
    Result;
END.
```

Bài toán 4.

Tính C_n^k

Chúng ta đã biết các công thức sau đây :

$C_n^k = 1$ (vớik=0 hoặc k=n)

$C_n^k = C^k_{n-1} + C^{k-1}_{n-1}$ (Với 0 < k < n)

Ta quen viết chương trình dạng sau :

```
Function C(k,n:integer):integer;
    Begin
        if (k=0)or(k=n) then C:=1
        else C:=C(k,n-1)+C(k-1,n-1);
```

End;

Chương Trình vẫn cho kết quả đúng nhưng xét về tính hiệu quả không đạt yêu cầu vì thời gian thực hiện sẽ rất lớn (Do tính lại nhiều lần một hay nhiều giá trị C(i,j) nào đó)

Có một biện pháp khắc phục đó là tính C_n^k theo công thức :

$$C_n^k = \frac{n!}{k!(n-k)!}$$

Nhưng ta cũng có thể giải bằng QHĐ .Dùng mảng C để lưu kết quả trung gian C[0..n,0..n]   (Với $C[i,j]= C_j^i$)

Mảng C như sau :

	k	0	1	2	3	4	5	..	...
n									
0		1							
1		1	1						
2		1	2	1					
3		1	3	3	1				
4		1	4	6	4	1			
5		1	5	10	10	5	1		
i		..	..	..	..	..	..	..	
N		...	...	...	...	...	...	...	...

Hình 5 Tam giác Pascal

Đoạn chương trình :

```
Fillchar(c,sizeof(c),0);
C[0,0]:=1;
For i:=2 to n do
        For j:=1 to i do
                C[i,j]:=C[i-1,j]+C[i-1,j-1];
```

Khi duyệt xong ta có kết quả là C[n,k].

Qua các ví dụ trên ta đã hiểu khá rỏ thế nào là **quy hoạch động** và làm thế nào để giải một bài toán bằng phương pháp **quy hoạch động** . Sau đây ta hãy cùng luyện tập quy hoạch động bằng cách giải các bài tập sau đây.

Bài 1:

Có n loại đồ vật,đồ vật thứ i có thể tích là v[i] và có giá trị là a[i].Cần xếp các đồ vật trên vào ba lô có thể tích V sao cho tổng giá trị các vật xếp vào là lớn nhất.

Giải quyết bài toán trong 3 trường hợp sau:

a./Mỗi loại có một đồ vật.

b./Mỗi loại co t[i] đồ vật cho trước.

c./Mỗi loại có số đồ vật không hạn chế.

Chương trình mẫu:

```
Program Balo;
     Uses Crt;
     Const
        Maxn=100;
        fin='BALO.INP';
        fout='BALO.OUT';
     Var  g,x:array[0..maxn,0..maxn] of integer;
        a,c:array[1..maxn] of byte;
        n,m:byte;
{---------------------------------}
Procedure Init;
      var f:text;
         i,j:byte;
      Begin
         assign(f,fin);
         reset(f);
         readln(f,n,m);
         for i:=1 to n do
            readln(f,a[i],c[i]);
         close(f);
         Fillchar(g,sizeof(g),0);
         for j:=1 to m do
            begin
            x[1,j]:=j div a[1];
            g[1,j]:=x[1,j]*c[1];
            end;
      End;
{--------------------------------------}
Procedure Slove;
      var k,v,u,xk:byte;
         max:integer;
      Begin
         for k:=2 to n do
            for v:=1 to m do
```

```
            for u:=0 to v do
            begin
                xk:=(v-u) div a[k];
                max:=g[k-1,u]+c[k]*xk;
                if max>g[k,v] then
                begin
                    g[k,v]:=max;
                    x[k,v]:=xk;
                end;
            end;
    End;
{-----------------------------------------}
Procedure Result;
    var k,v,i:byte;
      f:text;
      need:array[1..maxn] of byte;
    Begin
        assign(f,fout);
        rewrite(f);
        writeln(f,g[n,m]);
        k:=n;
        v:=m;
        Repeat
            need[k]:=x[k,v];
            v:=v-need[k]*a[k];
            dec(k);
        Until k<1;
        for i:=1 to n do write(f,need[i],' ');
        close(f);
    End;
{-------------------------------------------}
BEGIN
    Init;
    Slove;
    Result;
END.
```

Bài 2:

Cho n loại tiền xu trị giá k[1],k[2],...,k[n] xu .Cần đổi T đồng(tiền giấy) ra tiền xu sao cho số xu cần dùng là ít nhất (cho biết 1 đồng bằng 100 xu).

Bài 3: Cho dãy Fibonaci:

$F_1 = F_2 = 1$

$F_n = F_{n-1} + F_{n-2}$

Tính Fn với $n \le 200$.

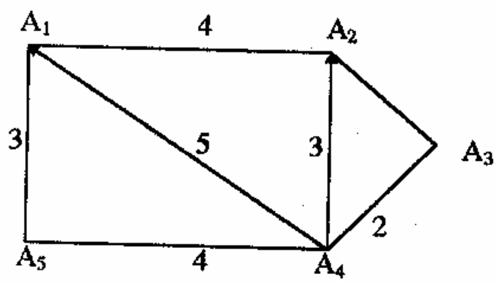
Bài 4:

Cho n thành phố , khoảng cách giữa hai thành phố i,j là L[i,j]. Tìm một đường đi qua n thành phố rồi trở về thành phố xuất phát sau cho tổng độ dài của cuộc hành trình là nhỏ nhất.

Bài 5: *Đề thi Tin Học Olympic 30-4-1997 tại Trường chuyên LÊ HỒNG PHONG (TP HCM)*

Cho đa giác n đỉnh $A_1, A_2, .., A_n$ .Hãy tìm cách chia đa giác trên thành các tam giác sao cho tổng độ dài các đường chia là nhỏ nhất.

Ví dụ:



Hình 6.

Bài 6: *Đề thi Học Sinh Giỏi Quốc Gia năm 1995-1996 Bảng A*

Với n nguyên dương ,xét tập $\mathcal{S}_n$ là tập tất cả các dãy số nguyên không âm $A=(a_0,a_1,..,a_{2k})$ thỏa mãn các điều kiện :

$1 \le k \le n$, (1)

$a_0=a_{2k}=0$, (2)

$|a_i-a_{i+1}|=1$, $i=0,1,2,..,2k-1$. (3)

Ta định nghĩa quan hệ thứ tự từ điển "<" trên tập $\mathcal{S}_n$ như sau :Hai dãy số $X(x_0,x_1,...,x_p)$ và $Y(y_0,y_1,...,y_q)$ thuộc tập $\mathcal{S}_n$ có quan hệ X<Y nếu tồn tại $0 \le r \le \min(p,q)$ sao cho $x_i = y_i$,$y \le r$ đồng thời r=p hoặc $x_{r+1}<y_{r+1}$.

Các dãy số trong tập $\mathcal{S}_n$ được sắp xếp theo thứ tự từ điển và được đánh số từ 1 trở đi

<u>Ví dụ:</u> với n=3 tập $\mathcal{S}_3$ gồm 8 phần tử và thứ tự của chúng như sau :

1 (0 1 0)	5 (0 1 2 1 0)
2 (0 1 0 1 0)	6 (0 1 2 1 0 1 0)
3 (0 1 0 1 0 1 0)	7 (0 1 2 1 2 1 0)
4 (0 1 0 1 2 1 0)	8 (0 1 2 3 2 1 0)

Yêu cầu :

a) Với n cho trước, tính tổng số các dãy thuộc tập $\mathcal{T}_n$;

b) Với số nguyên m cho trước tìm dãy có thứ tự từ điển là m trong tập $\mathcal{T}_n$;

c) Với một dãy cho trước thuộc tập $\mathcal{T}_n$ xác định thứ tự từ điển của nó.

Dữ liệu vào: File văn bản BL4.INP

Dòng thứ nhất: n (n nguyên dương , $n \leq 46$),

Các dòng tiếp theo : mỗi dòng có thể có một trong hai dạng

1 m

hoặc 2 k a_0 a_1 .. a_{2k}

Các số trên một dòng cách nhau ít nhất một dấu cách.

Dạng đầu là dữ liệu cho câu hỏi b),dạng sau là dữ liệu cho câu hỏi c). Tổng số các dòng cho các câu hỏi loại b) , c) là không quá 50.

Kết quả :đưa ra file văn bản BL4.OUT:

Dòng đầu : số nguyên cho biết tổng số các dãy trong tập $\mathcal{T}_n$.

Các dòng sau : mỗi dòng tương ứng với một dòng yêu cầu trong dữ liệu vào.Với câu hỏi b) kết quả là dãy a_0 a_1 ... a_{2k},các số đưa ra trên một dòng , cách nhau ít nhất một dấu cách,qui ước ghi số 0 nếu không có dãy thỏa mãn điều kiện ra.Với câu hỏi c) kết quả là số nguyên m,đưa ra trên một dòng.

Đoạn chương trình mẫu:

của bạn *Bùi Thế Duy (Lớp chuyên Toán - Tin ĐHQG Hà Nội -Đội tuyển Tin Học QG 1996)*

```
program bl4;

uses crt;

const inputfilename='bl4.inp';
    outputfilename='bl4.out';
    limit=46;
    coso=1000000000;

type Tnum=array[1..3] of longint;
   Tset=record
       num:integer;
       value:array[0..2*limit] of byte;
      end;

var fin,fout:text;
```

```
  sum:array[0..2*limit,0..limit] of Tnum;
  {sum[i,j]: tong cac so co vi tri la i ma o vi tri do co gia tri la j}
  n:integer;
  a:tset;
  loc:Tnum;

procedure add(x1,x2:tnum;var x3:tnum);
 var t:longint;
 begin
  t:=x1[3]+x2[3];
  if t>coso then
   begin
    x3[3]:=t-coso;
    t:=1;
   end
  else
   begin
    x3[3]:=t;
    t:=0;
   end;
  t:=t+x1[2]+x2[2];
  if t>coso then
   begin
    x3[2]:=t-coso;
    t:=1;
   end
  else
   begin
    x3[2]:=t;
    t:=0;
   end;
  x3[1]:=t+x1[1]+x2[1];
 end;

procedure maken;
 var i,j:integer;
 begin
 {khoi tao}
 fillchar(sum,sizeof(sum),0);
```

```
  sum[2*n,0,3]:=1;
  sum[2*n,0,2]:=0;
  sum[2*n,0,1]:=0;
  {xong khoi tao}

  for i:=2*n-1 downto 0 do
   begin
    for j:=0 to n do
     begin

      sum[i,j,3]:=0;
      sum[i,j,2]:=0;
      sum[i,j,1]:=0;
      if (not odd(i))and(i>0)and(j=0) then sum[i,j,3]:=1;

      if j>0 then add(sum[i+1,j-1],sum[i,j],sum[i,j]);
      if j<n then add(sum[i+1,j+1],sum[i,j],sum[i,j]);
     end;
   end;
 end;

procedure tassign;
 begin
  assign(fin,inputfilename);
  assign(fout,outputfilename);
 end;

procedure solve;
 var t:integer;
    s:string;
 begin
  clrscr;
  reset(fin);
  rewrite(fout);
  readln(fin,n);
  maken;
  if sum[0,0,1]>0 then write(fout,sum[0,0,1]);
  if sum[0,0,2]>0 then write(fout,sum[0,0,2]);
```

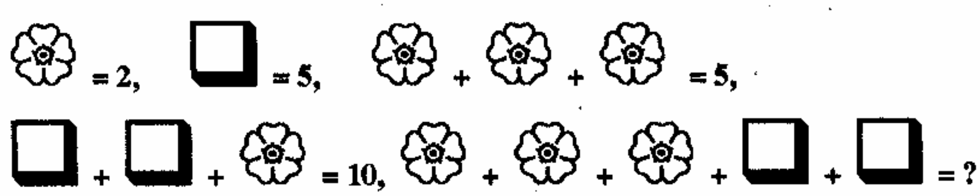
```
 writeln(fout,sum[0,0,3]);
end;
 close(fin);
 close(fout);
end;

begin
 tassign;
 solve;
end.
```

46
2 46 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Bài 7: *Đề thi Tin Học Quốc Tế năm 1995 tại Hà Lan*

SHOPPING



Hình 7.

Trong một cửa hàng mỗi loại hàng có một giá . ví dụ giá một bông hoa là 2 ICU (ICU-Đơn vị tiền) và giá một cái bình là 5 ICU .Để thu hút nhiều khách hàng ,cửa hàng đề ra một số cách bán đặc biệt.

Một cách bán đặc biệt liên quan đến việc bán một hoặc một số mặt hàng với một giá chung được giảm.Ví dụ:3 bông hoa bán với giá 5 ICU thay vì 6 ICU ;2 bình với một bông hoa bán với giá 10 ICU thay vì 12 ICU.

Viết chương trình tính giá mà một khách hàng trả cho một nhu cầu mua hàng để tận dụng một cách tối ưu các cách bán đặc biệt-có nghĩa là chi phí thấp nhất có thể được, Ví dụ với các giá và cách bán đặc biệt như trên,giá (thấp nhất) để mua được 3 hoa và 2 bình là 14 ICU:2 bình và một hoa với giá được giảm là 10 ICU,2 hoa với giá bình thường là 4 ICU.

Dữ liệu vào : gồm hai file INPUT.TXT và OFFER.TXT .file thứ nhất mô tả nhu cầu mua .file thứ hai mô tả các cách bán đặc biệt.trong cả 2 file chỉ có các số nguyên.

dòng thứ nhất của file INPUT.TXT chứa số b là số loại hàng cần mua (0≤ b ≤5). Mỗi dòng trong số b dòng tiếp theo chứa 3 số c,k,p . Giá trị c (1 ≤ c ≤ 999) là mã của loại hàng (hai loại hàng khác nhau có mã khác nhau).Giá trị k là số đợn vị hàng (với mã c) cần mua (1 ≤ k ≤ 5).Giá trị p là giá bình thương của một đơn vị hàng (với mã c) 1 ≤ p ≤ 999.Chú ý rằng trong một yêu cầu có không quá 5x5=25 đơn vị hàng .

Dòng thứ nhất của file OFFER.TXT chứa số s là số cách bán đặc biệt (0≤ s ≤99) .Mỗi dòng trong số s dòng tiếp theo mô tả một cách bán đặc biệt bằng cách cho cấu trúc và giá (đã được giảm) của nó.Số đầu tiên n của một dòng như vậy là số loại hàng trong cách bán đặc biệt tương ứng với dòng đó,(1 ≤ n ≤ 5) n cặp số tiếp theo (c,k) trong đó c là mã loại hàng,k là số đơn vị loại hàng đó (1 ≤ k ≤ 5,1 ≤ c ≤ 999).Số p cuối cùng trong dòng là giá (đã được giảm) của lô hàng này(1 ≤ p ≤ 9999).giá đã được giảm nhỏ hơn tổng các giá bình thường.

bạn không được phép mua thêm các loại hàng không cần mua cho dù việc đó làm giảm chi phí.

Dữ liệu ra:ghi vào file OUTPUT.TXT , một dòng chi phí thấp nhất phải trả cho nhu cầu mua nêu trong file vào.

Ví dụ về dữ liệu vào và ra:

Mã của hoa là 7 mã của bình là 8.

INPUT.TXT
2 7 3 2 8 2 5

OFFER.TXT
2 1 7 3 5 2 7 1 8 2 10

OUTPUT.TXT
14

Chương trình mẫu :

```
Program BaiSHOPPING;
    Uses Crt;
    Const
      fin1='INPUT.TXT';
      fin2='OFFER.TXT';
      fout='OUTPUT.TXT';
      maxb=5;
      maxc=99;
      maxname=999;
    Type index=0..maxb;
    Var  name:array[1..maxb] of integer;
      thu :array[1..maxname] of byte;
      need:array[1..maxb] of byte;
      p   :array[1..maxb] of word;
```

```
        db  :array[1..maxc,1..maxb] of byte;
        tdb :array[1..maxc] of word;
        tan :array[index,index,index,index,index] of word;
        s,b:byte;
{---------------------------------}
Procedure Init;
      var i:integer;
        n,j,so:byte;
        ten:integer;
        f:text;
      Begin
        fillchar(need,sizeof(need),0);
        assign(f,fin1);
        reset(f);
        readln(f,b);
        for i:=1 to b do
          begin
          readln(f,name[i],need[i],p[i]);
          thu[name[i]]:=i;
          end;
        close(f);
        assign(f,fin2);
        reset(f);
        readln(f,s);
        for i:=1 to s do
          begin
          read(f,n);
          for j:=1 to n do
            begin
              read(f,ten,so);
              db[i,thu[ten]]:=so;
            end;
          readln(f,tdb[i]);
          end;
        close(f);
      End;
{-----------------------------------------}
Procedure Slove;
      Var i1,i2,i3,i4,i5:byte;
```

```
    j:integer;
    min,m:word;
  Begin
    fillchar(tan,sizeof(tan),0);
    for i1:=0 to need[1] do
      for i2:=0 to need[2] do
        for i3:=0 to need[3] do
          for i4:=0 to need[4] do
            for i5:=0 to need[5] do
    begin
      min:=i1*p[1]+i2*p[2]+i3*p[3]+i4*p[4]+i5*p[5];
      for j:=1 to s do
        if (i1>=db[j,1])and(i2>=db[j,2])and(i3>=db[j,3])and
          (i4>=db[j,4])and(i5>=db[j,5]) then
          begin
            m:=tan[i1-db[j,1],i2-db[j,2],i3-db[j,3],
                i4-db[j,4],i5-db[j,5]]+tdb[j];
            if m<min then min:=m;
          end;
      tan[i1,i2,i3,i4,i5]:=min;
    end;
  End;
{--------------------------------------}
Procedure Result;
  var f:text;
  begin
    assign(f,fout);
    rewrite(f);
    writeln(f,tan[need[1],need[2],need[3],need[4],need[5]]);
    close(f);
  end;
{------------------------------------}
BEGIN
  Init;
  Slove;
  Result;
END.
```

Bài 8: *Đề thi Tin Học Quốc Tế năm 1996 tại Hungari*

PREFIX

Đoạn đầu dài nhất (Longest Prefix)

Cấu trúc của một số vật thể sinh học được biểu diễn bởi một dãy các thành phần của chúng.Các thành phần này được kí hiệu bởi các chữ cái hoa.Các nhà sinh học quan tâm tới việc phân rả một dãy dài thành các dãy ngắn hơn.Các dãy ngắn đó được gọi là các dãy nguyên thủy.Ta nói rằng dãy S có thể ghép được từ một tập cho trước P các dãy nguyên thủy nếu tìm được n dãy nguyên thủy $p_1,...p_n$ thuộc P sao cho ghép của chúng $p_1,..,p_n$=S.Việc ghép các dãy nguyên thủy $p_1,..,p_n$ có nghĩa là đặt chúng liên tiếp liền nhau theo thứ tự đó.Một dãy nguyên thủy có thể có mặt nhiều lần trong phép ghép và không nhất thiết mọi dãy nguyên thủy phải có mặt.Chẳng hạn dãy ABABACABAAB có thể ghép được từ tập các dãy nguyên thủy

{A,AB,BA,CA,BBC}

k kí tự đầu của một dãy S được gọi là đoạn đầu với độ dài k của S.Viết chương trình với dữ liệu vào là một tập P các dãy nguyên thủy và một dãy T các thành phần. Chương trình tính độ dài của đoạn đầu dài nhất của dãy T mà đoạn đầu đó có thể ghép được từ các dãy nguyên thủy trong P.

Dữ liệu vào

Dữ liệu vào trong 2 file. File INPUT.TXT mô tả tập các dãy nguyên thủy P còn file DATA.TXT chứa dãy T cần được xem xét.Dòng thứ nhất của file INPUT.TXT chứa n là số lượng dãy nguyên thủy trong P ($1 \le n \le 100$). Mổi dãy nguyên thủy được cho bởi hai dòng liên tiếp.Dòng thứ nhất chứa độ dài L của dãy nguyên thủy($1 \le L \le 20$) Dòng thứ hai chứa một xâu có độ dài L chứa các chữ cái hoa (từ "A" đến "Z"). n dãy nguyên thủy này từng đôi một khác nhau.

Mổi dòng của file DATA.TXT chứa một cái hoa ở vị trí đầu tiên.File này kết thúc bằng dòng chứa một dấu chấm ('.') ở vị trí đầu tiên.

Độ dài của dãy từ 1 đến 500000.

Dữ liệu ra

Viết vào dòng thứ nhất của file OUTPUT.TXT độ dài của đoạn đầu dài nhất của T mà đoạn đầu đó có thể ghép được từ tập P.

Ví dụ về dữ liệu vào và ra

INPUT.TXT
5
1
A
2
AB
3
BBC
2
CA
2
BA

DATA.TXT
A
B
A
B
A
C
A
B
A
A
B
C
B
.

OUTPUT.TXT
11

Chương trình mẫu :

```
Program Longest_Prefix;
    Uses Crt;
    Const
        maxin=100;
        maxil=20;
        fin1='INPUT.TXT';
        fin2='DATA.TXT';
        fout='OUTPUT.TXT';
    Type
        typep=string[maxil+1];
    Var p:array[1..maxin] of Typep;
        l:array[1..maxin] of byte;
        truocok:array[0..maxil] of boolean;
        pos,lb:longint;
        n,maxl:byte;
        c:Typep;
{-----------------------------------}
Function Thoa:boolean;
    var i:byte;
    Begin
        thoa:=true;
```

```
        for i:=1 to maxl do
            if truocok[i] then exit;
        thoa:=false;
    End;
{----------------------------------}
Procedure ReadandSolve;
    var f:text;
        i:byte;
        x:char;
        ht:boolean;
    Begin
        assign(f,fin1);
        reset(f);
        readln(f,n);
        maxl:=0;
        for i:=1 to n do
            begin
            readln(f,l[i]);
            if l[i]>maxl then maxl:=l[i];
            readln(f,p[i]);
            end;
        close(f);
        Fillchar(truocok,sizeof(truocok),true);
        truocok[0]:=true;
        assign(f,fin2);
        reset(f);
        pos:=0;
        c:='                    ';
        while (not eof(f))and thoa do
            Begin
                readln(f,x);
                if x<>'.' then
                begin
                inc(pos);
                for i:=1 to length(c)-1 do
                    c[i]:=c[i+1];
                c[length(c)]:=x;
                ht:=false;
                for i:=1 to n do
```

```
                    if (p[i]=copy(c,length(c)-l[i]+1,l[i]))
                      and truocok[l[i]] and (pos>=l[i]) then
                      begin
                      ht:=true;
                      break;
                      end;
                  truocok[0]:=ht;
                  for i:=maxl downto 1 do truocok[i]:=truocok[i-1];
                  end;
               End;
           Close(f);
           lb:=pos;
           while not truocok[pos-lb] do
           dec(lb);
           if not truocok[0] then inc(lb);
        End;
{-----------------------------------}
Procedure Result;
        var f:text;
        Begin
           assign(f,fout);
           rewrite(f);
           writeln(f,lb);
           close(f);
        End;
{-----------------------------------}
BEGIN
   ReadandSolve;
   Result;
END.
```

Tài liệu tham khảo:

- Tuyển tập 50 Đề thi Tin Học Quốc Gia và Quốc Tế.
- Giáo trình "Giáo trình Tối ưu hóa chương trình" của PGS-PTS Bùi Minh Trí PTS Bùi Thế Tâm.
- Các vấn đề lập trình của Trần Đức Huyên.