**8.5. SHELLSORT**

Nhược điểm của thuật toán sắp xếp kiểu chèn thể hiện khi mà ta luôn phải chèn một khóa vào vị trí gần đầu dãy. Trong trường hợp đó, người ta sử dụng phương pháp ShellSort.

Xét dãy khoá: k1, k2, …, kn. Với một số nguyên dương h: 1  h  n, ta có thể chia dãy đó thành h dãy con:

Dãy con 1: k1, k1+h, k1 + 2h, …

Dãy con 2: k2, k2+h, k2 + 2h, …

…

Dãy con h: kh, k2h, k3h, …

*Ví dụ như dãy (4, 6, 7, 2, 3, 5, 1, 9, 8); n = 9; h = 3. Có 3 dãy con.*

Dãy các khoá chính: 4 6 7 2 3 5 1 9 8

Dãy con 1: 4 2 1

Dãy con 2: 6 3 9

Dãy con 3: 7 5 8

Những dãy con như vậy được gọi là dãy con xếp theo độ dài bước h. Tư tưởng của thuật toán ShellSort là: Với một bước h, áp dụng thuật toán sắp xếp kiểu chèn từng dãy con độc lập để làm mịn dần dãy khoá chính. Rồi lại làm tương tự đối với bước h div 2 … cho tới khi h = 1 thì ta được dãy khoá sắp xếp.

Như ở ví dụ trên, nếu dùng thuật toán sắp xếp kiểu chèn thì khi gặp khoá k7 = 1, là khoá nhỏ nhất trong dãy khoá, nó phải chèn vào vị trí 1, tức là phải thao tác trên 6 khoá đứng trước nó. Nhưng nếu coi 1 là khoá của dãy con 1 thì nó chỉ cần chèn vào trước 2 khoá trong dãy con đó mà thôi. Đây chính là nguyên nhân ShellSort hiệu quả hơn sắp xếp chèn: Khoá nhỏ được nhanh chóng đưa về **gần** vị trí đúng của nó.

procedure ShellSort;

var

i, j, h: Integer;

tmp: TKey;

begin

h := n div 2;

while h <> 0 do {Làm mịn dãy với độ dài bước h}

begin

for i := h + 1 to n do

begin {Sắp xếp chèn trên dãy con ai-h, ai, ai+h, ai+2h, …}

tmp := ki;

j := i - h;

while (j > 0) and (kj > tmp) do

begin

kj+h := kj;

j := j - h;

end;

kj+h := tmp;

end;

h := h div 2;

end;

end;