1.- ***Explicar detalladamente el método de clasificación por montón describiendo la definición de montón de Williams, la manera de construir un monton "in situ" de Floyd y el algoritmo de clasificación. Usar para ello el siguiente arreglo inicial:***

44	*55*	*12*	*42*	*94*	*18*	*6*	*67*

Completar el programa siguiente (en Modula-2) para que realice el desplazamiento de Floyd de un montón.

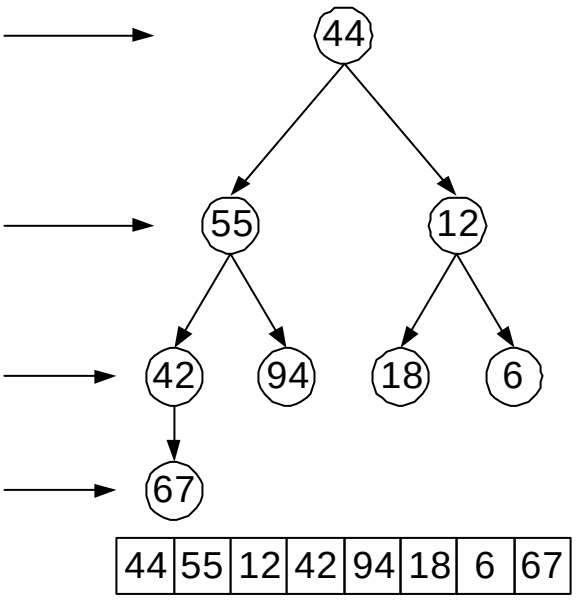
```
PROCEDURE sift(L,R : index);
  VAR i,j : index;
  x :       item;
BEGIN
  i:=L;j:=2*L;x:=a[L];
  IF (j<R)& … THEN j:=j+1 END;
  WHILE (j<=R)& … . DO
    … ; I:=j ; …;
    IF (j<R) & … THEN j:=j+1 END;
  END;
END sift.
```

Un montón se define como una secuencia de llaves h_L , h_{l+1} ....., h_R tales que:

- $h_i \leq h_{2i}$ y $h_i \leq h_{2i+1}$ para = L.....R/2
- h_1 = mín(h_1,h_2,....,h_n)

<u>Construcción de un montón in situ de Floyd:</u>

Para comprender la generación y funcionamiento del montón de Floyd, es conveniente utilizar la analogía gráfica con un árbol. Si consideramos la tabla de valores indicada, y la colocamos en un árbol, por el orden que se dan y suponiendo que se leen horizontalmente se puede entender fácilmente el funcionamiento de Floyd.



Los elementos h_m....h_n forman ya un montón con m=(n DIV 2) + 1, ya que no tienen “ramas” por debajo

Para el resto de los elementos se comienza un proceso iterativo que consiste en añadir un nuevo elemento y a continuación recolocar sus ramas, esto es, cuando existen en sus ramas elementos menores, ir intercambiandolo con ellos hasta que alcance su posición definitiva.

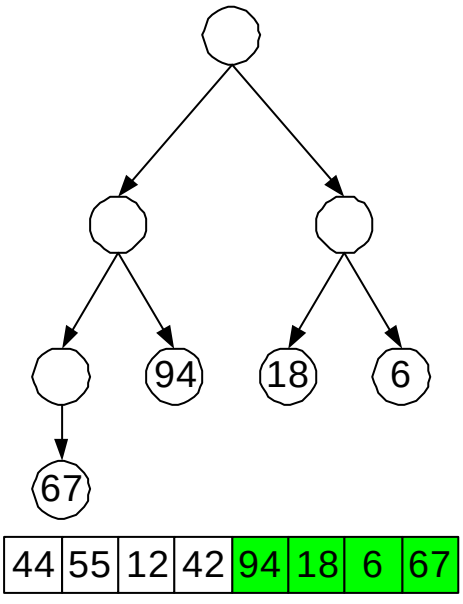
La secuencia de elementos ordenados se indicará mediante un sombreado, destacando mediante una línea punteada los intercambios necesarios. Se muestra a continuación la creación del montón.

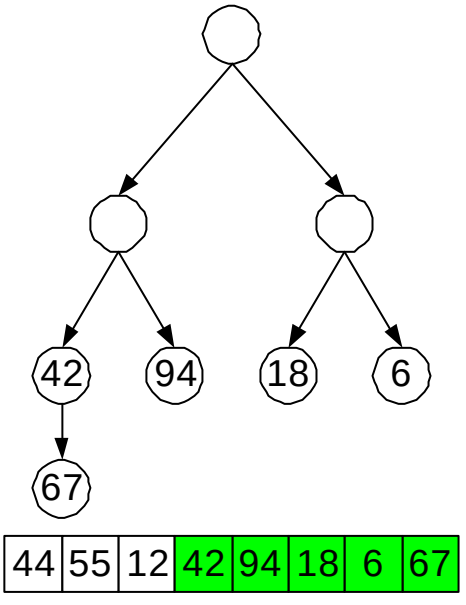
```
L:= (n DIV 2) + 1;
R:=n;
WHILE  L >  1 DO
  L:=L - 1;
  Sift(L,R);
END;
```

Previamente resulta conveniente haber completado la función “sift”.

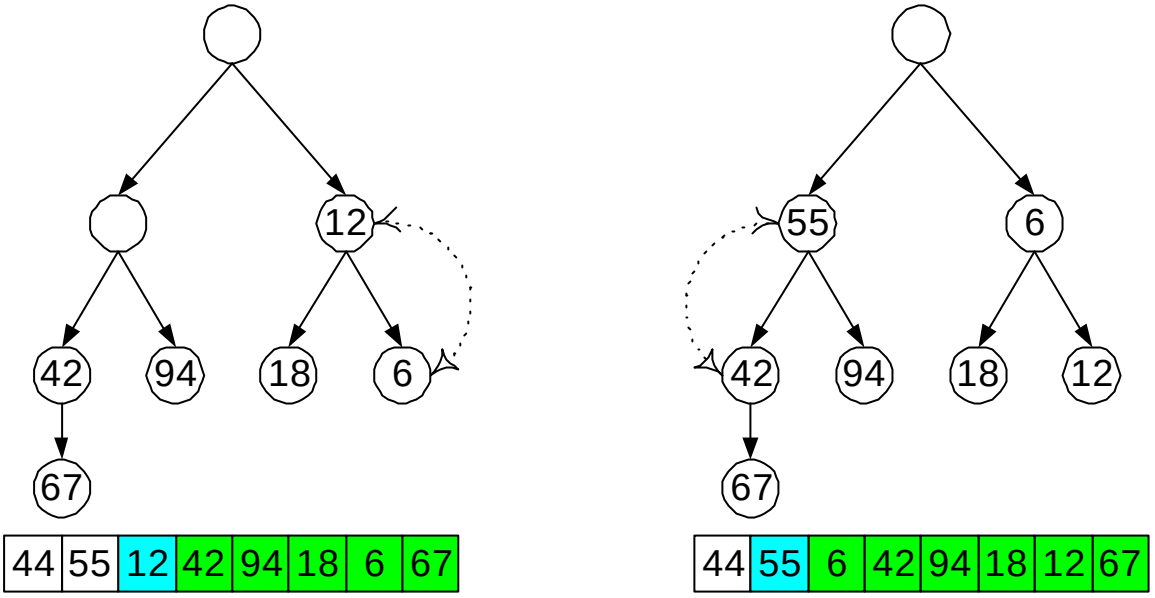
```
Algoritmo de desplazamiento de Floyd:
PROCEDURE Sift(L,R:index);
VAR i,j:index;
      x:item;
BEGIN
  i:=L;j:=2*L;x:=a[L];
  IF (j < R) & (a[j] < a[j+1] THEN j:=j+1 END;
  WHILE (j <= R) & (x < a[j] DO
      a[i] := a[j];
       i := j;
       j := 2 * j;
      IF (j < R) & (a[j] < a[j+1] THEN j:=j+1 END;
  END;
END Sift;
```

Como se había indicado, los elementos h_m....h_n forman ya un montón, siendo m=8/2+1=5, puesto que no tienen ramas que partan de ellos. Al el elemento “42”, no se hace necesario ningún tipo de intercambio por el mismo motivo.

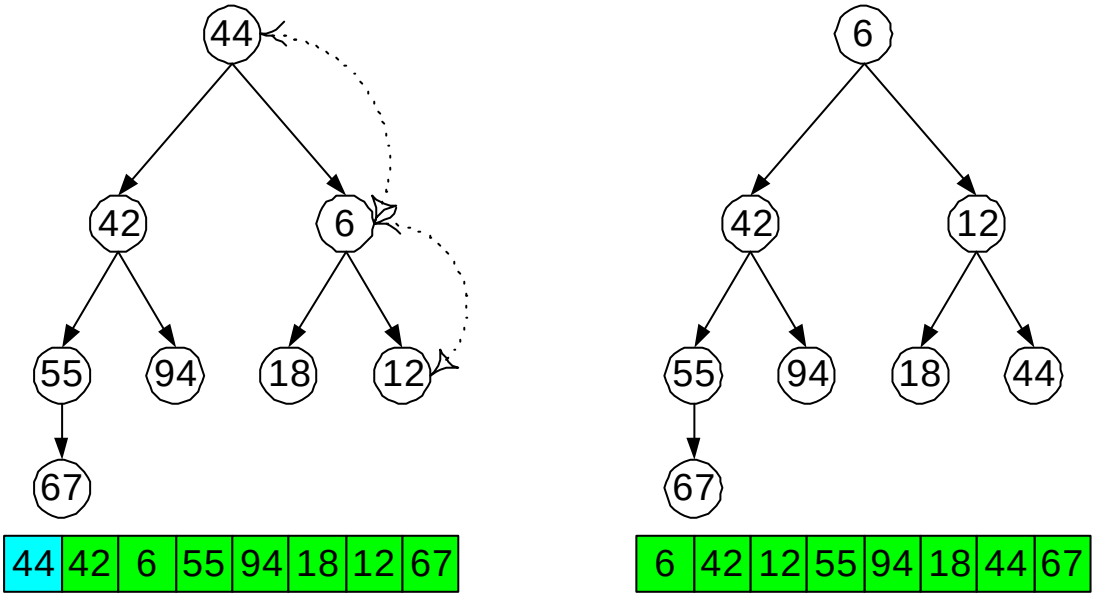




Al añadir el valor “12”, tenemos que el elemento “6” está en su rama, y al ser menor deben intercambiarse. Lo mismo sucede al añadir el valor “55” con el elemento “42”.



Tras colocar de nuevo los elementos del árbol, se añade el último elemento. En este caso, el valor “44” se debe intercambiar primero con el “6” y posteriormente con el “12”.



Si representamos de nuevo linealmente la secuencia de valores, recordando que primero se añade el elemento y posteriormente se realizan los intercambio, tendremos:

44	55	12	42	94	18	6	67
44	55	12	42	94	18	6	67
44	55	6	42	94	18	12	67
44	42	6	55	94	18	12	67
6	42	12	55	94	18	44	67

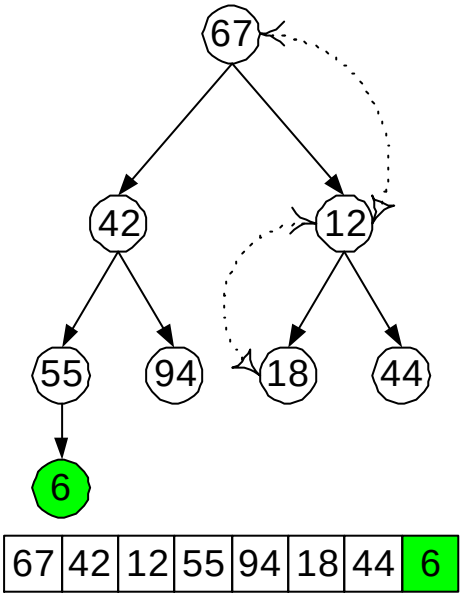
Clasificación del montón:

En la clasificación del montón se sigue un procedimiento análogo. Se comienza intercambiando el primero del montón por el último de la secuencia, y a continuación se reordena el montón obtenido (ahora con un elemento menos). El algoritmo que realiza la clasificación del montón es:

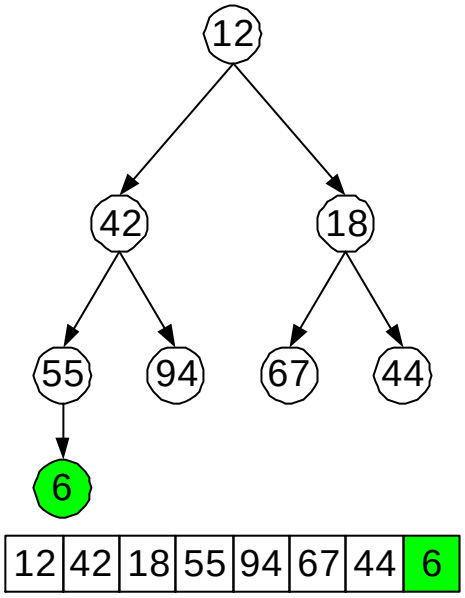
```
WHILE R > 1 DO
  x:=a[1];          Intercambia el primero
  a[1:=a[R];        con el último sin clasificar.
  a[R]:=x;
  R:=R-1;
  Sift(L,R);        Hace el desplazamiento (clasificación) por las ramas del árbol.
END;
```

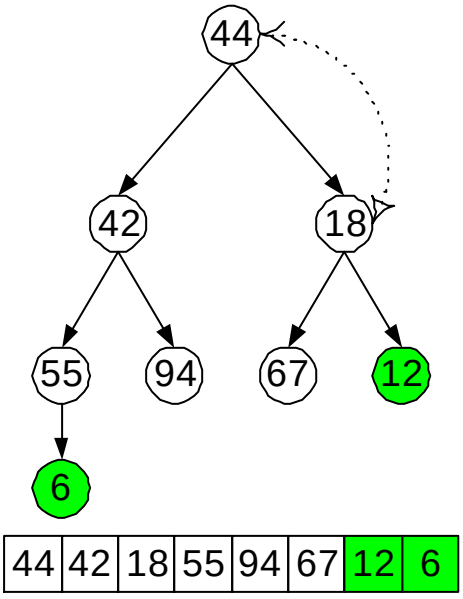
Utilizando de nuevo la analogía del árbol, y separando la fase de intercambio y de reordenación, tenemos.

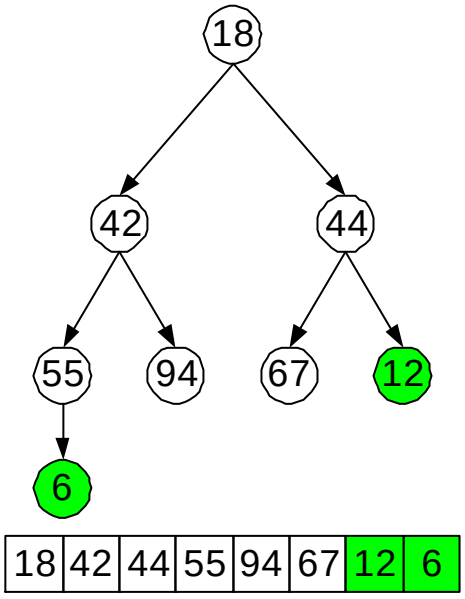
Intercambio

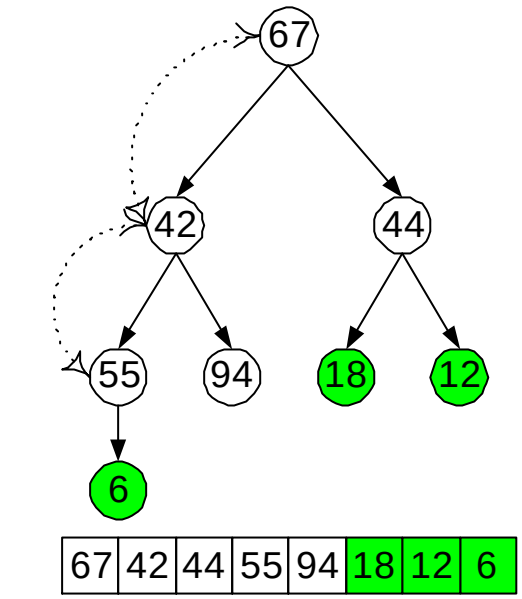


Reordenación

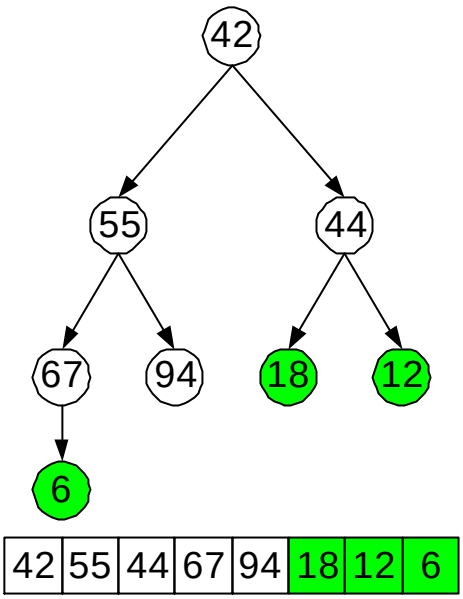




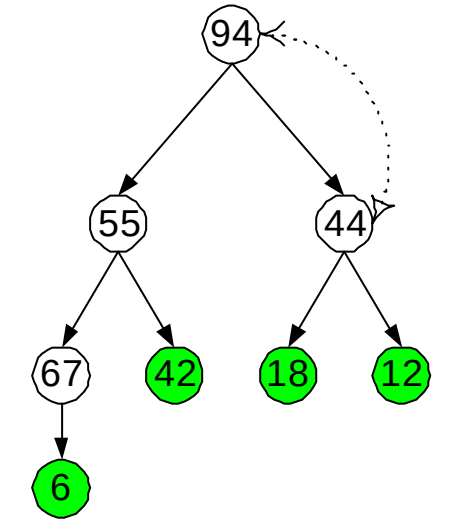




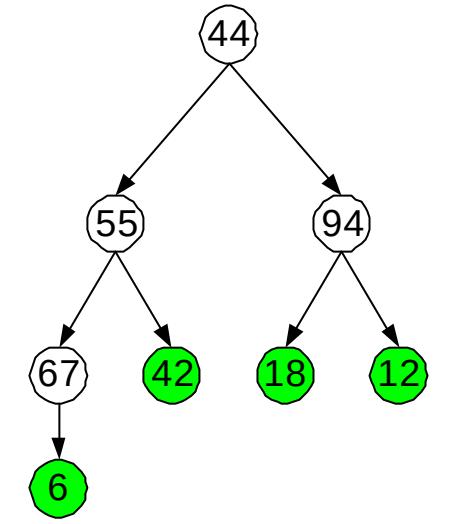
67	42	44	55	94	18	12	6



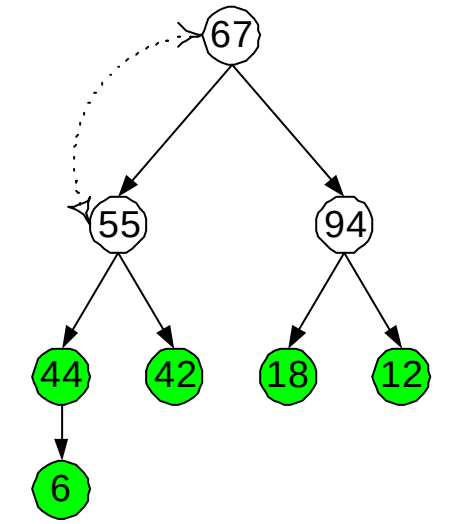
42	55	44	67	94	18	12	6



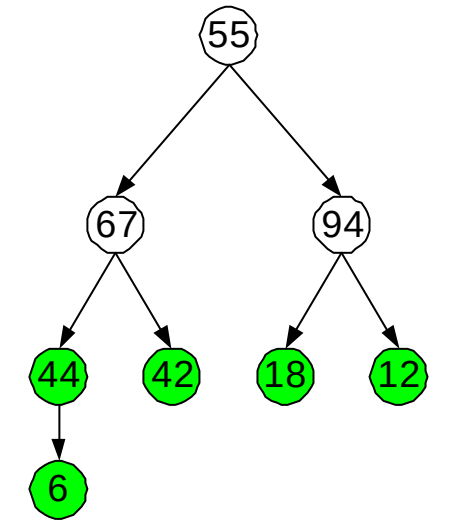
94	55	44	67	42	18	12	6



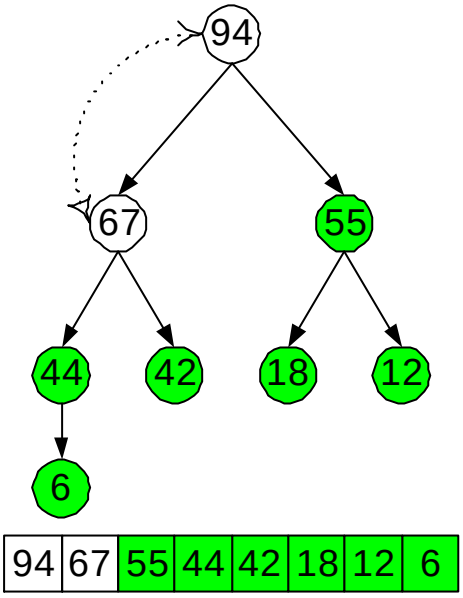
44	55	94	67	42	18	12	6

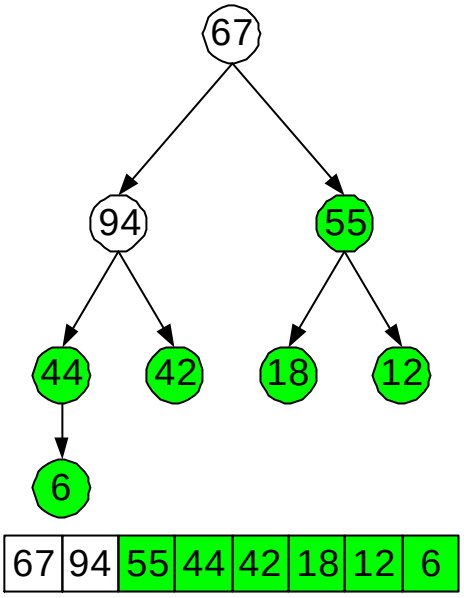


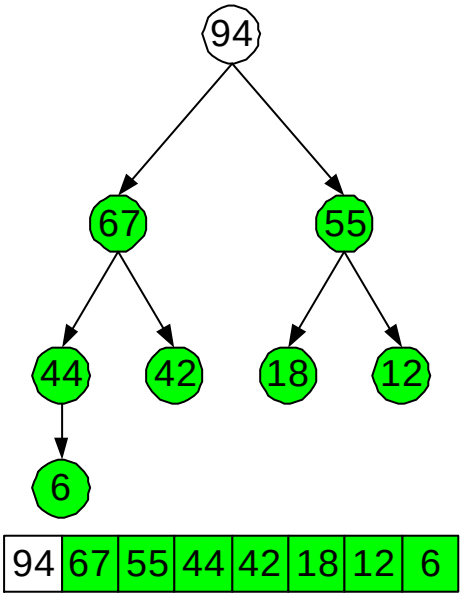
67	55	94	44	42	18	12	6



55	67	94	44	42	18	12	6







Tomando de nuevo los elementos por niveles, y escribiendo los pasos realizados, se tiene

6	42	12	55	94	18	44	67
12	42	18	55	94	67	44	6
18	42	44	55	94	67	12	6
42	55	44	67	94	18	12	6
44	55	94	67	42	18	12	6
55	67	94	44	42	18	12	6
67	94	55	44	42	18	12	6
94	67	55	44	42	18	12	6

La clasificación por montón de Floyd no da buenos resultados, ya que primero corre a la izquierda los elementos grandes antes de ponerlos en su sitio. Su eficacia aumenta al aumentar n.

El caso peor necesita del orden de $n\log(n)$ movimientos, mientras que si el arreglo inicial está en ordenado no se necesita ningún movimiento para realizar el montón, reduciéndose las operaciones a las de clasificación; por tanto, los movimientos se reducen a $n\log(n)/2$.