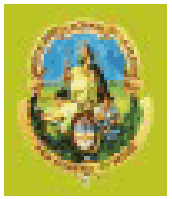


Algoritmos y Estructuras de Datos

Cursada 2014

Prof. Catalina Mostaccio
Prof. Alejandra Schiavoni

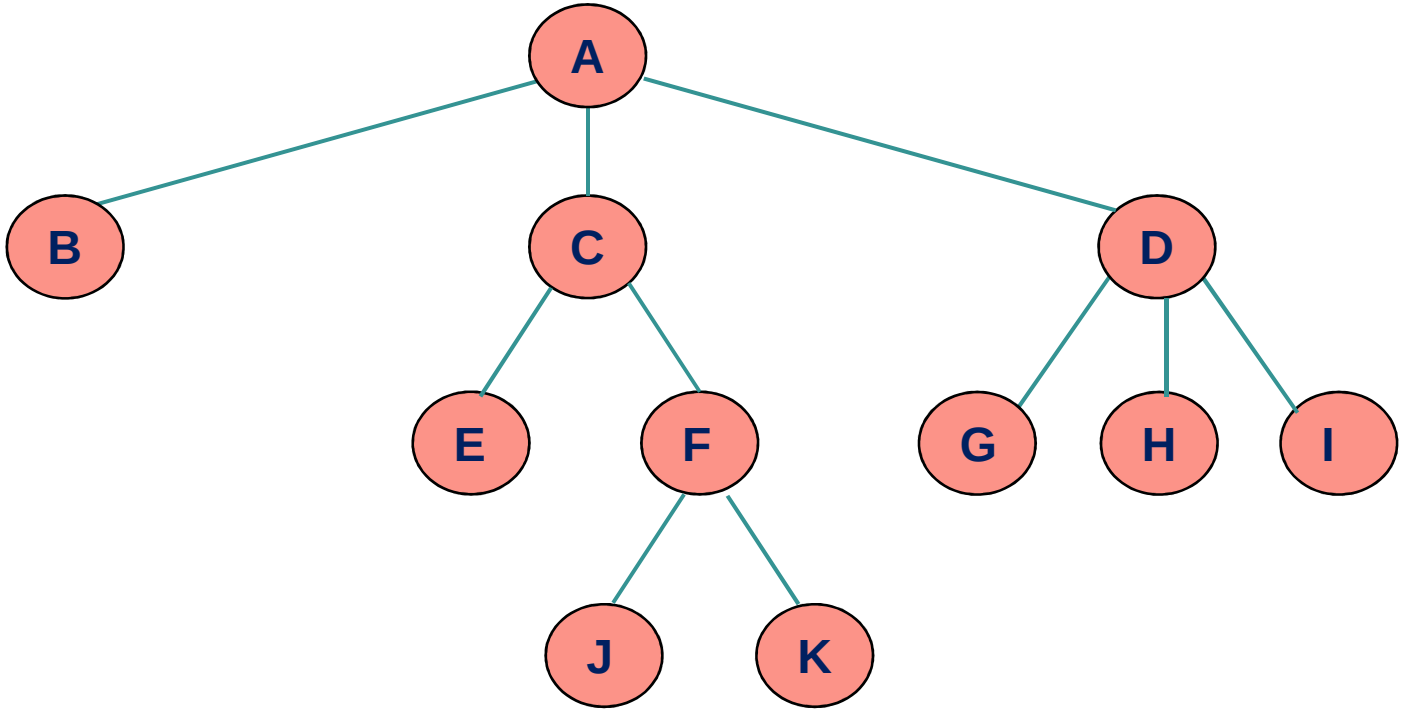
Facultad de Informática - UNLP

Árboles Generales

Ejercicios

1) ¿Cuántos niveles tiene el árbol?
2) ¿Cuántos nodos hay en cada nivel del árbol?
3) ¿Cuántos nodos hay en el nivel *k* del árbol?

Resolución de Ejercicios



Recorrido por niveles

```
Seudocódigo Recorrido_Niveles {
      q: cola de vértices;
      encolar raíz R en q;
      mientras cola no se vacíe {
         desencolar v de q;
         imprimir (dato de v);
         para cada hijo w de v
                encolar w en q;
      }
}
```

A B C D E F G H I J K

Ejercicio

¿Cuántos niveles tiene el árbol?

- En el recorrido por niveles no se distingue a qué nivel pertenece cada nodo
- Necesito llevar un control de los niveles

Inserto una marca al finalizar cada nivel

Resolución del Ejercicio 1

```
Seudocódigo Ejerc1-Niveles {
    q: cola de vértices;
    encolar raíz R en q;
    mientras cola no se vacíe {
        desencolar v de q;
        imprimir (dato de v);
        para cada hijo w de v
          encolar w en q; }
}
```

encolar ?? en q

if cont. de **v** es ?? then encolar ?? en q

Ojo ! Podría imprimir ??

Resolución del Ejercicio 1

```
Seudocódigo Ejerc1-Niveles {
    q: cola de vértices;
    encolar raíz R en q;
    mientras cola no se vacíe {
       desencolar v de q;
```

encolar ?? en q

if cont. de **v** es ?? then encolar ?? en q

else

```
       imprimir (dato de v);
       para cada hijo w de v
          encolar w en q; }
```

Resolución del Ejercicio 1

```
Seudocódigo Ejerc1-Niveles {
    q: cola de vértices;
    encolar raíz R en q; encolar ?? en q;
     mientras cola no se vacíe {
        desencolar v de q;
        if cont. de v es ??
            else
                if cont. de v no es ?? then
        imprimir (dato de v);
        para cada hijo w de v
            encolar w en q; }
    }
```

& q no vacía;

then encolar ?? en q

A B C D E F G H I J K

Resolución del Ejercicio 1

```
Seudocódigo Ejerc1-Niveles {
   q: cola de vértices;
    encolar raíz R en q; encolar ?? en q;
    mientras cola no se vacíe {
       desencolar v de q;
       si v es ?? & q no está vacía then
                encolar ?? en q;
       else
                si v no es ?? then {
                      imprimir (dato de v);
                      para cada hijo w de v
                            encolar w en q; }
    }
}
```

nroNiveles = 0;

nroNiveles ++;

return nroNiveles;

Resolución del Ejercicio 2

```
Seudocódigo Ejerc2-Niveles {
    q: cola de vértices;
    encolar raíz R en q; encolar ?? en q;
    mientras cola no se vacíe {
       desencolar v de q;
       si v es ?? & q no está vacía then
                encolar ?? en q;
       else
                si v no es ?? then {
                   imprimir (dato de v);
                   para cada hijo w de v
                            encolar w en q; }
    }
}
```

cantNodos: array de enteros;

nroNivel = 0;

nroNivel ++;

cantNodos[nroNivel]++;

Resolución del Ejercicio 3

```
Seudocódigo Ejerc3-Niveles (int k) {
    q: cola de vértices;
    encolar raíz R en q; encolar ?? en q;
    mientras cola no se vacíe {
       desencolar v de q;
       si v es ?? & q no está vacía then
                encolar ?? en q;
       else
                si v no es ??    then {
                   imprimir (dato de v);
                   para cada hijo w de v
                            encolar w en q;
                }
    }
}
```

nroNivel = 0;

cantNodos = 0;

nroNivel ++;

& nroNivel == k;

Contar la cantidad de nodos en el nivel **k**

return cantNodos;

Resolución del Ejercicio 3

```
Seudocódigo Ejerc3-Niveles (int k) {
    q: cola de vértices; nroNivel=0; cantNodos=0;
     encolar raíz R en q; encolar ?? en q;
     mientras cola no se vacíe {
       desencolar v de q;
        si v es ?? & q no está vacía then
                  encolar ?? en q;
                  nroNivel++;
        else
                  si v no es ?? & nroNivel==k then {
                    mientras cola no se vacíe & v no es ??
                    {cantNodos++;
                     desencolar v de q;}
                  }
     }
   return cantNodos;
}
```