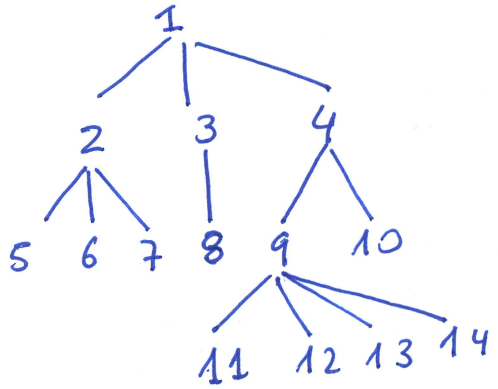
LECCION 23: Arboles Generales

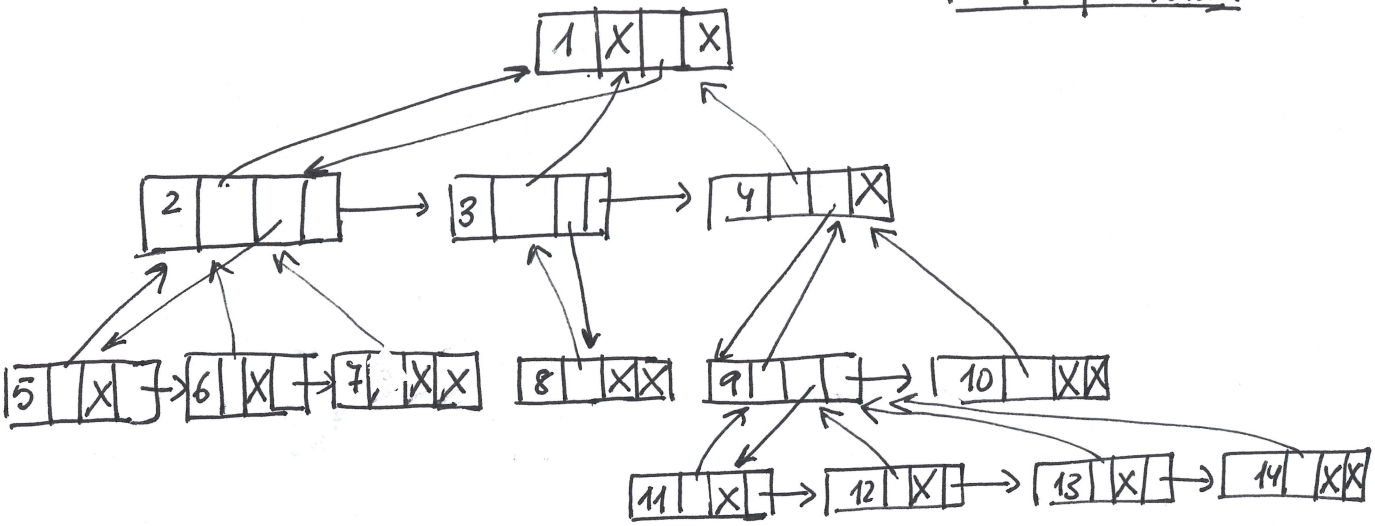
Pueden contener cualquier numero de hijos



Representacion

```
template <class T>
struct info-nodo{
    T et;
    info-nodo *padre, *hizq, *hermadrcha;
};
```

info-nodo




```
template <class T>
info-nodo <T>*  CrearRaiz(const T & e){
        info-nodo <T> * n = new info-nodo <T>{
        n->et=e;
        n->padre=n->hizq=n->hermadrcha=0;
        return n;
}
template <class T>
void InsertarHijoIzquierda(                    info-nodo<T> * n,
                           info-nodo <T>* & T2){
    if (T2!=0){
        T2->hermadrcha = n->hizq;
        T2->padre = n;
        n->hizq=T2;
        T2=0; //
    };
}
```

LECCION 23: Arboles Generales.

```
template <class T>
void InsetwHermanoDrcha ( info-nodo<T>*n,
                          info-nodo <T>* dT2) {

      if (T2!=0){
          T2 -> hermadrcha = n->hermadrcha;
          T2 -> padre=n->padre;
          n-> hermadrch = T2;
          T2=0;
      }
}

template <class T>
info-nodo<T> * PodarHijo Izqda ( info-nodo <T>*n){
      info-nodo <T> * res=0;
      if (n->hizq != 0){
          Res=n->hizq;
          n->hizq = n->hizq -> hermadrch;
          Res ->padre = res->hermadrch =0;
      }
      return res;
}

template <class T>
info-nodo <T> * Podar HermaDrcha( info-nodo<T>*n)
      info-nodo<T> * res=0;
      if (n-> hermadrcha!=0){
          res=n->hermadrcha;
          n-> hermadrcha = res->hermadrcha;
          res->padre =res->hermadrch=0;
      }
      return res;
}
```

3 Arboles
Generales

LECCION 23: Arboles Generales

```
template <class T>
int Altura (info_nodo <T> * T) {
    if (T==0)
        return -1
    else{
        int max=-1;
        info_nodo <T> *aux;
        for (aux= T->hizq; aux!=0; aux=aux->hermadrcha)
        {   if (Altura (aux) >max)
                max = Altura(aux);
        }
        return 1+max;
    }
}

template <class T>
void Destruir (info_nodo <T>*t){
    if (t!=0){
        Destruir(t->hizq);
        Destruir (t->hermadrch);
        delete t;
    }
}

template <class T>
void copiar (info_nodo <T> * s, info_nodo <T> * &d){
        if (s==0){
            d=0;
        }
        else{
            d= CrearRaiz(s->et);
            copiar(s->hizq, d->hizq);
            copiar(s->hermadrcha, d->hermadrcha);
            if (d->hizq!0)
                d->hizq->padre=d;
            if (d->hermadrch!=0)
                d->hermadrch->padre=d->padre;
}
}
```