Per cominciare,
importiamo un paio di pacchetti che saranno utili durante lo svolgimento delle analisi proposte :

```
with(combinat)
```

```
[Chi, bell, binomial, cartprod, character, choose, composition, conjpart, decodepart, encodepart,
    eulerian1, eulerian2, fibonacci, firstcomb, firstpart, firstperm, graycode, inttovec, lastcomb,
    lastpart, lastperm, multinomial, nextcomb, nextpart, nextperm, numbcomb, numbcomp,
    numbpart, numbperm, partition, permute, powerset, prevcomb, prevpart, prevperm,
    randcomb, randpart, randperm, rankcomb, rankperm, setpartition, stirling1, stirling2,
    subsets, unrankcomb, unrankperm, vectoint]
```

(1)

```
with(combstruct)
```

```
[agfeqns, agfmomentsolve, agfseries, allstructs, count, draw, finished, gfeqns, gfseries, gfsolve,
    iterstructs, nextstruct]
```

(2)

Prima analisi : *verifica del risultato del numero di alberi s − ari dato il numero di nodi interni.*

```
s := 5
```

$$5 \tag{3}$$

```
TreeS := {T = Union(L, Prod(N, seq(T, i = 1 ..s))), L = Epsilon, N = Atom}
```

$$\{L = \mathrm{E},\ N = Atom,\ T = Union(L, Prod(N, T, T, T, T, T))\} \tag{4}$$

Testiamo ora la definizione di albero s − ario esplicitata poco sopra,
utilizzando la funzione draw che estrae una permutazione a caso :

```
draw([T, TreeS], size = 5)
```

```
Prod(N, Prod(N, L, L, Prod(N, L, L, L, L, L), Prod(N, L, L, L, L, L), Prod(N, L, L, L, L, L)), L,
    L, L, L)
```

(5)

Calcoliamo il numero totale di strutture possibili :

```
count := numelems(allstructs([T, TreeS], size = 5)) :
```

```
count
```

$$2530 \tag{6}$$

Ora calcoliamo il valore teorico tramite la formula proposta dalla teoria :

```
val_teorico := proc(s, n)
 1/((s − 1)n + 1)·binomial(s·n, n)
end proc;
```

proc(s, n) *combinat*:-binomial$(s * n, n) / ((s - 1) * n + 1)$ **end proc** **(7)**

```
val_teorico(5, 5)
```

$$2530 \tag{8}$$

Come si evince i due risultati coincidono.

Proponiamo quindi anche una procedura che,
dato un valore k (pari al numero di simulazioni da condurre, ovvero il numero di alberi da creare),
restituisce il numero totale di foglie presenti all'interno di tutti i k alberi s − ari creati :

```
countLeaves := proc(k)
 local i, acc, tree, numFoglie;
 acc := 0;
```

```
 for i from 1 to k do
   tree := draw( [T, TreeS], size = 5 );
   numFoglie := numboccur(tree, Prod(N, L, L, L, L, L) );
   acc := acc + numFoglie;
 od;
 return acc;
 end proc;
```

```
proc(k)                                                              (9)
    local i, acc, tree, numFoglie;
    acc := 0;
    for i to k do
        tree := draw( [T, TreeS], size = 5 );
        numFoglie := numboccur(tree, Prod(N, L, L, L, L, L) );
        acc := acc + numFoglie
    end do;
    return acc
end proc
```

Richiamiamo la procedura precedente con $k = 10000$ *(ovvero con* 10000 *simulazioni di alberi s − ari) e salviamo il risultato all'interno di una variabile* :

```
tot_leaves_s := countLeaves(10000)
                                19098                                (10)
```

Seconda analisi : *verifica del numero di foglie negli alberi binari dato il numero di nodi interni.*

Il numero di nodi foglia negli alberi binari tende a $\left(\frac{n}{4}\right)$ *quando* $n \to inf$

Definiamo la struttura degli alberi binari :

```
Tree2 := {T = Union(L, Prod(N, T, T) ), L = Epsilon, N = Atom}
                  {L = E, N = Atom, T = Union(L, Prod(N, T, T) ) }   (11)
```

Definiamo ora una procedura che, dati due parametri k e n, tramite simulazione ritorna il valore totale delle foglie presenti all'interno di k alberi binari con n nodi interni :

```
countLBin :=proc(k, n)
 local i, acc, tree, numFoglie;
 acc := 0;
 for i from 1 to k do
   tree := draw( [T, Tree2], size = n );
   numFoglie := numboccur(tree, Prod(N, L, L) );
   acc := acc + numFoglie;
 od;
 return acc;
 end proc;
```

```
proc(k, n)                                                           (12)
    local i, acc, tree, numFoglie;
```

```
    acc := 0;
    for i to k do
        tree := draw( [T, Tree2], size = n );
        numFoglie := numboccur(tree, Prod(N, L, L) );
        acc := acc + numFoglie
    end do;
    return acc
end proc
```

Ora richiamiamo la procedura con $k = 10000$ *(numero di alberi) e* $n = 100$ *(numero di nodi interni di ogni albero) :*

$$tot\_leaves := countLBin(10000, 100)$$

$$253651 \qquad \textbf{(13)}$$

Ora prendiamo il valore ritornato, lo dividiamo per il numero di alberi su cui abbiamo effettuato la simulazione, e notiamo che il risultato tende a $\left(\frac{n}{4} \right)$ *:*

$$avg\_leaves := evalf\left(\frac{tot\_leaves}{10000} \right)$$

$$25.36510000 \qquad \textbf{(14)}$$

In questo caso n era uguale a 100*, quindi correttamente si ottiene un valore attorno a* 25.

Terza analisi : verifica del numero di foglie negli alberi planari dato il numero di nodi interni.
Il numero di nodi foglia degli alberi planari tende a $\left(\frac{n}{2} \right)$.

Definiamo ora una procedura che, dati due parametri k e n, tramite simulazione ritorna il valore totale delle foglie presenti all 'interno di k alberi planari con n nodi interni :

```
countFPlan :=proc(k, n)
 local i, acc, tree, numProd, TreeP;
 acc := 0;
 TreeP := {T = Union(N, seq(Prod(N, seq(T, i = 1 ..j) ), j = 1 ..n − 1) ), L = Epsilon, N = Atom};
 for i from 1 to k do
  tree := draw( [T, TreeP], size = n );
  numProd := numboccur(tree, Prod);
  acc := acc + (n − numProd);
 od;
acc;
end proc;
```

```
proc(k, n)                                                    (15)
    local i, acc, tree, numProd, TreeP;
```

```
    acc := 0;
    TreeP := {L = Epsilon, N = Atom, T = Union(N, seq(Prod(N, seq(T, i = 1 ..j)), j = 1
    ..n − 1))};
    for i to k do
        tree := draw([T, TreeP], size = n);
        numProd := numboccur(tree, Prod);
        acc := acc + n − numProd
    end do;
    acc
end proc
```

Ora richiamiamo la procedura con i valori di $k = 10000$ (*ovvero* 10000 *alberi*) *e n* $= 100$ (*alberi con* 100 *nodi interni ciascuno*) *e salviamo il risultato del totale delle foglie all'interno di una variabile* :

$$tot\_leaves\_plan := countFPlan(10000, 100)$$

$$499945 \tag{16}$$

$$avg\_leaves\_plan := evalf\left(\frac{tot\_leaves\_plan}{10000}\right)$$

$$49.99450000 \tag{17}$$

In questo caso n è 100,

e quindi il risultato ottenuto di circa 50 *nodi foglia rispecchia il risultato teorico di* $\left(\frac{n}{2}\right)$.

Quarta analisi : *numero medio di foglie negli alberi binari di ricerca dato il numero di nodi interni.*

Il numero medio di foglie all'interno degli alberi binari di ricerca tende a $\left(\frac{n}{3}\right)$.

Siccome gli alberi binari di ricerca non possiedono una struttura definita a priori e regolare come nel caso degli alberi binari standard, o degli alberi s, definiamo una procedura che, data una permutazione di m elementi, restituisce una struttura ad albero corrispondente all'albero binario di ricerca costruito su quella specifica permutazione. Ciò ci permetterà di calcolare, analogamente a quanto effettuato per le altre analisi, il numero medio di foglie e di confrontarlo con il risultato teorico atteso.

```
drawBinS :=proc(val, subtree)
 local new_node, new_sx_node, new_dx_node;
 global Tree, depth;
  if (numboccur(subtree, L) = 1) then
    new_node := draw([T, Tree2], size = 1);
    new_node := subs(N = val, new_node);
    Tree := subsop(depth = new_node, Tree);
    depth := [ ];
    return;
  else:
    if (val ≤ op(1, subtree)) then
      depth := [op(depth), 2];
      drawBinS(val, op(2, subtree));
```

```
    else:
      depth := [op(depth), 3];
      drawBinS(val, op(3, subtree));
    end if;
  end if;
 end proc;
proc(val, subtree)                                                    (18)
    local new_node, new_sx_node, new_dx_node;
    global Tree, depth;
    if numboccur(subtree, L) = 1 then
        new_node := draw([T, Tree2], size = 1);
        new_node := subs(N = val, new_node);
        Tree := subsop(depth = new_node, Tree);
        depth := [ ];
        return
    else
        if val <= op(1, subtree) then
            depth := [op(depth), 2]; drawBinS(val, op(2, subtree))
        else
            depth := [op(depth), 3]; drawBinS(val, op(3, subtree))
        end if
    end if
end proc
```

Definiamo una struttura base di un albero che sarà la struttura di partenza per la costruzione degli alberi binari di ricerca :

```
Tree := draw([T, Tree2], size = 0)
```

$$L \qquad (19)$$

Ora creiamo una lista, al momento vuota, che conterrà le "coordinate" del nuovo nodo da inserire all'interno dell'albero :

```
depth := [ ];
```

$$[\ ] \qquad (20)$$

Definiamo ora una procedura che,
dato un valore k pari al numero di simulazioni da condurre (ovvero il numero di alberi da creare) e un valore n (ovvero il numero di nodi interni), restituisce il numero totale di foglie presenti all'interno di ogni albero.
Tramite questo valore, come per le altre analisi, potremo facilmente calcolare il numero medio di nodi foglia all'interno degli alberi binari di ricerca creati :

```
countAbr :=proc(k, n)
 local arr, i, j, acc, foglia, numFoglie;
 global depth, Tree;
 acc := 0;
 for i from 1 to k do
  arr := randperm(n);
  Tree := draw([T, Tree2], size = 0);
  depth := [ ];
```

```
    for j from 1 to n do
      drawBinS(arr[j], Tree);
    end do;
    for j from 1 to n do
      acc := acc + numboccur(Tree, Prod(j, L, L));
    end do;
   end do;
  acc;
end proc;
```

```
proc(k, n)                                                    (21)
    local arr, i, j, acc, foglia, numFoglie;
    global depth, Tree;
    acc := 0;
    for i to k do
        arr := combinat:-randperm(n);
        Tree := draw([T, Tree2], size = 0);
        depth := [ ];
        for j to n do drawBinS(arr[j], Tree) end do;
        for j to n do
            acc := acc + numboccur(Tree, Prod(j, L, L))
        end do
    end do;
    acc
end proc
```

Richiamiamo la procedura con $k = 10000$ *(ovvero* 10000 *alberi) e* n $= 100$ *(ovvero* 100 *nodi interni per ogni albero) e lo assegnamo ad una variabile* :

$$tot\_leaves\_abr := countAbr(10000, 100)$$

$$336673$$

$$avg\_leaves\_abr := evalf\left(\frac{tot\_leaves\_abr}{10000}\right)$$

$$33.66730000 \qquad (23)$$

Come possiamo notare, il risultato ottenuto rispecchia il valore teorico, in quanto si ha che per n $= 100$ *il numero di nodi foglia dovrebbe risultare circa* 33,
valore ritrovato tramite questa simulazione.