# Disjunktne množice (angl. *Disjoint sets*)

# ADT Disjoint sets

- Množico elementov želimo razbiti na disjunktne podmnožice glede na neko relacijo med elementi

- Gradimo množice od spodaj navzgor

  1)Vsak element je ena podmnožica;

  2)Manjše podmnožice združujemo v večje podmnožice, če so elementi iz ene in druge podmnožice v dani relaciji;

- Za vsak element moramo vedeti, kateri podmnožici pripada

# ADT Disjoint sets

**MAKENULL(S)** generira prazno množico množic *S*.

**MAKESET(x, S)** tvori novo množico $\{x\}$ in jo doda v *S*.

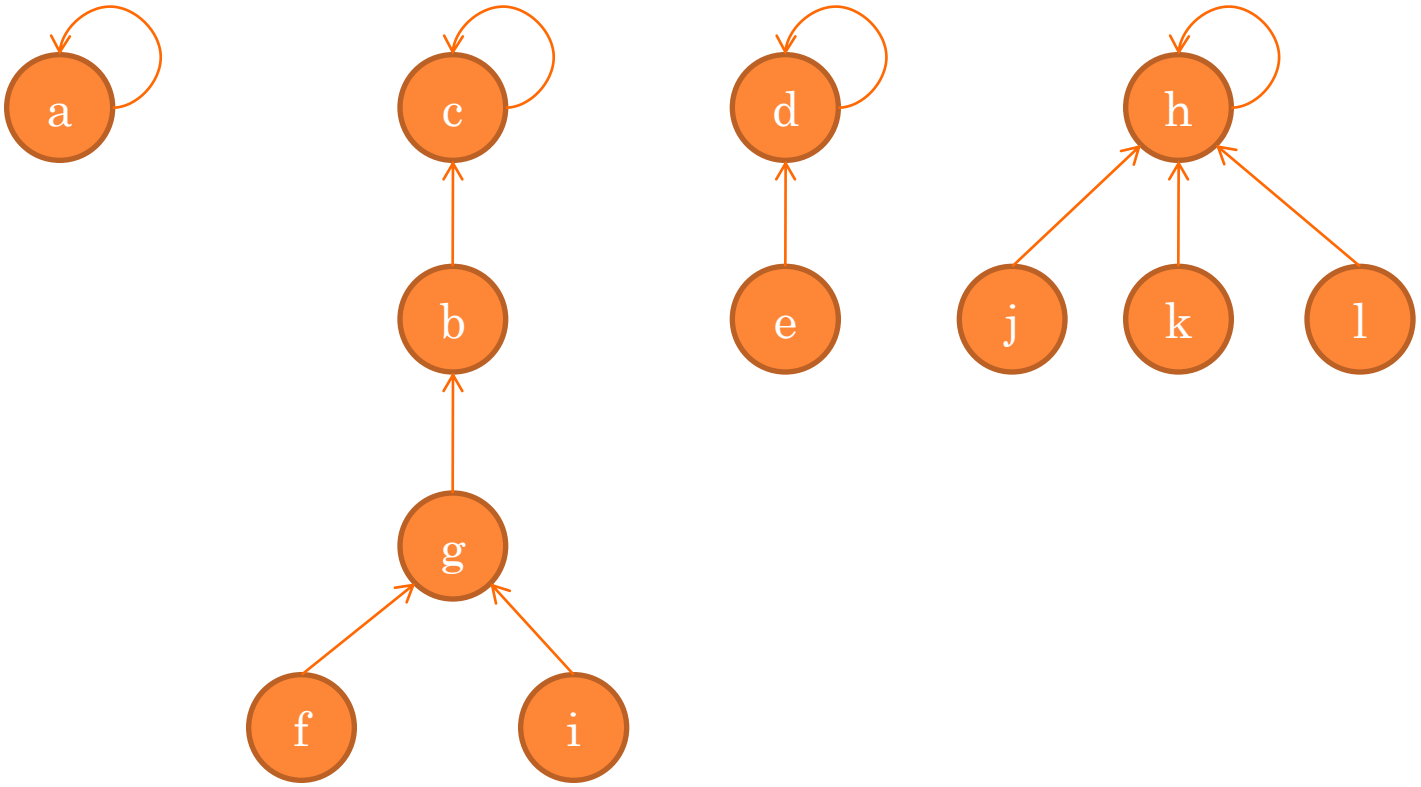
**UNION(A1, A2, S)** združi dve disjunktni podmnožici $A_1$ in $A_2$ v novo podmnožico.

**FIND(x, S)** vrne podmnožico, katere element je *x*.

```
public interface DisjointSet {
   public abstract void makenull() ;
   public abstract DisjointSubset makeset(Object x) ;
   public abstract void union(DisjointSubset a1, DisjointSubset a2) ;
   public abstract DisjointSubset find(DisjointSubset x) ;
} // interface DisjointSet
```

# Implementacija z Gozdom

- Vsaka množica je drevo
- Vsak element kaže na očeta v drevesu
- Koren kaže sam nase
- Množica je identificirana s korenom
- Gozd disjunktnih množic: { {a}, {b, c, f, g, i}, {d, e}, {h, j, k, l} }

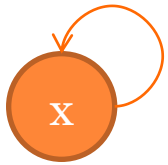


# Implementacija z Gozdom

Za učinkovito implementacijo vozlišče potrebuje št. elementov poddrevesa:

```
public class DisjointSubset {
   Object value ;
   DisjointSubset parent ;
   int noNodes ; // moc podmnozice
} // class DisjointSubset
```

Operacija MAKESET iz enega elementa *x* tvori množico *{x}*



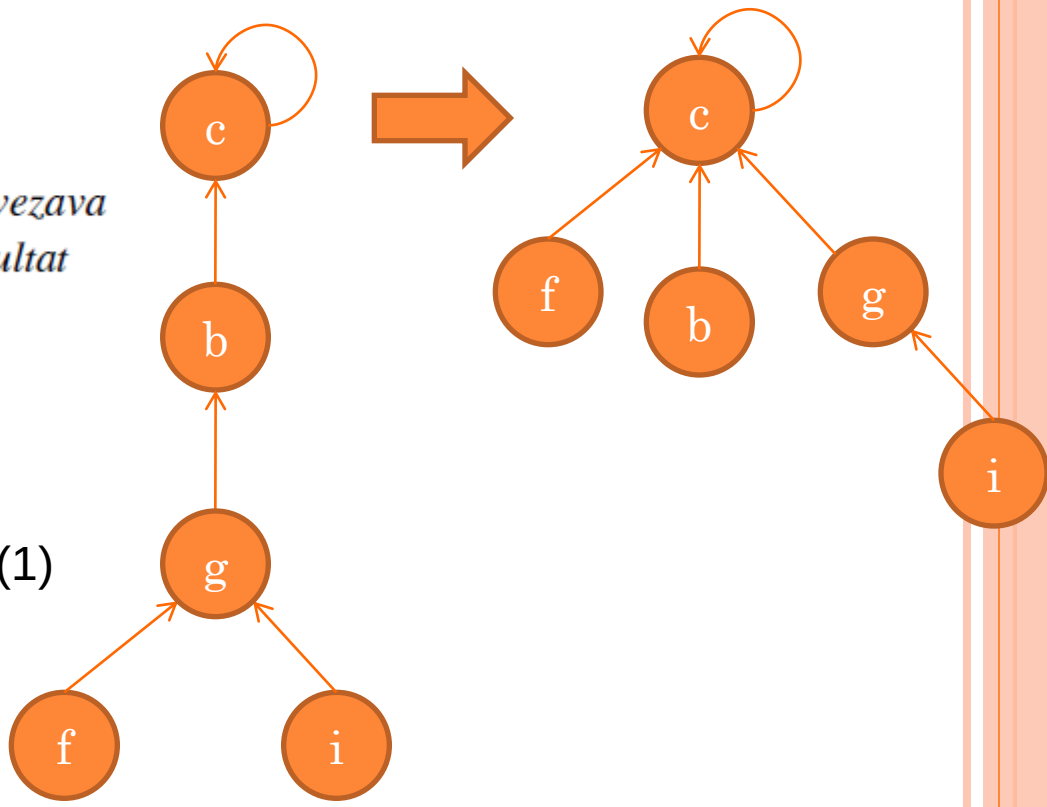

```
public DisjointSubset makeset(Object x) {
    DisjointSubset newEl = new DisjointSubset() ;
    newEl.value = x ;
    newEl.noNodes = 1 ;
    newEl.parent = newEl ;
    return newEl ;
}
```

Časovna zahtevnost: *O(1)*

# Implementacija z Gozdom

FIND(x): vrne množico (t.j. koren drevesa), ki ji pripada element *x.*
Če je drevo izrojeno, je plezanje do korena lahko reda O(n).
Da se izrojenosti na dolgi rok izognemo,
vsa vozlišča na poti prevežemo na koren:

```
public DisjointSubset find(DisjointSubset x) {
  if (x == x.parent)
    return x ;
  else {
    x.parent = find(x.parent) ; // prevezava
    return x.parent ; // in hkrati rezultat
  }
} // find
```

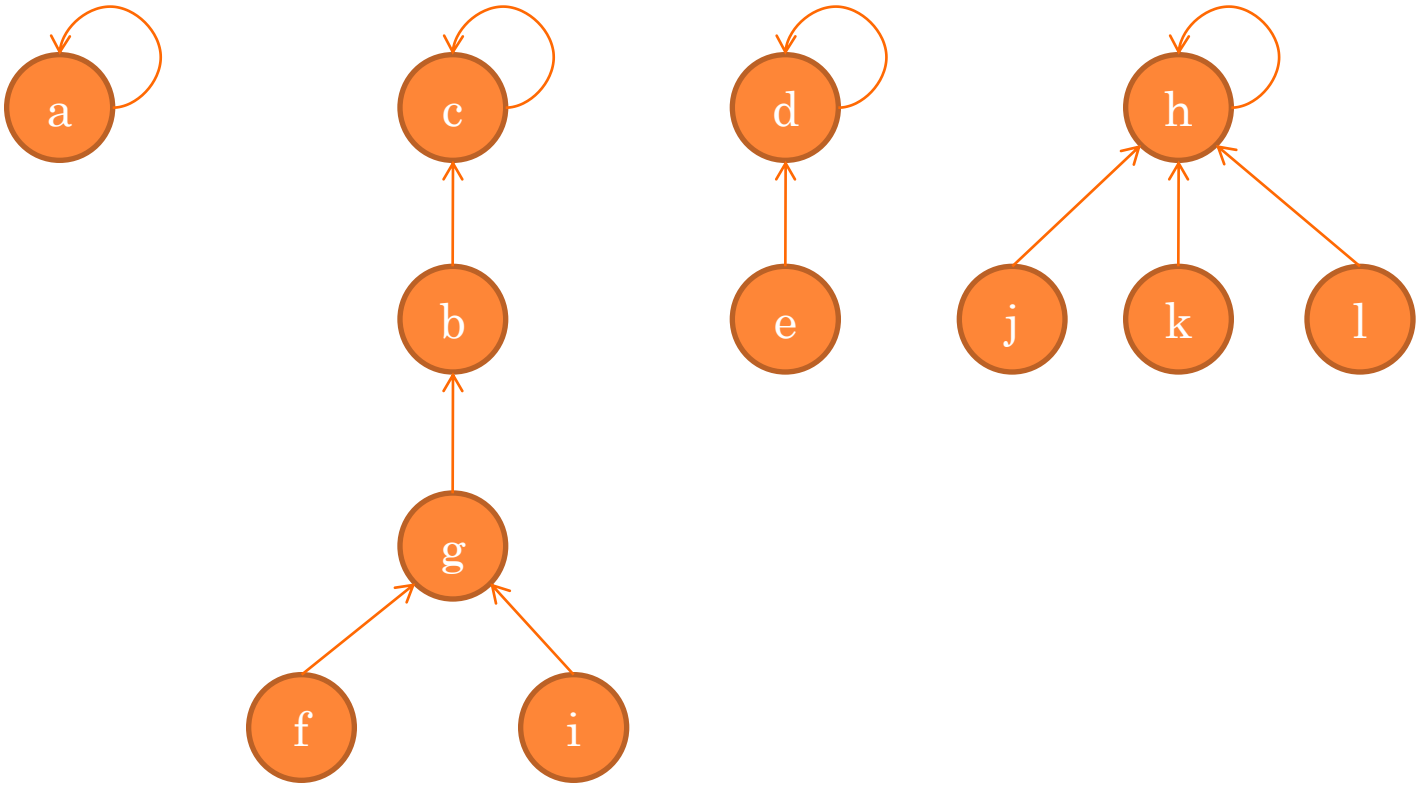



FIND(f):
O(n) → O(1)

# Implementacija z Gozdom

UNIJA: koren ene podmnožice prevežemo na koren druge.
Ker želimo čimmanj izrojeno drevo,
vedno prevežemo manjšo množico na večjo.

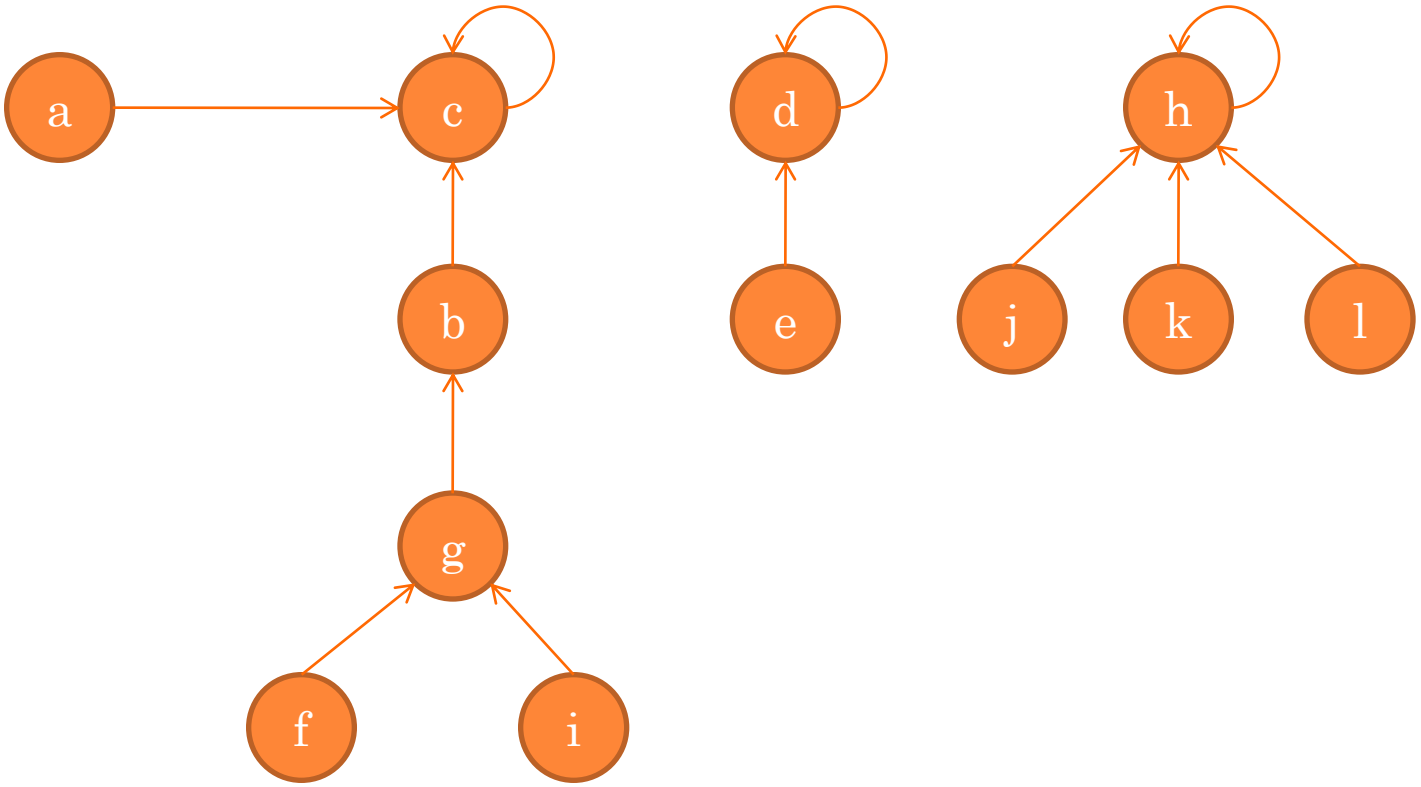
union(a,c)



# Implementacija z Gozdom

UNIJA: koren ene podmnožice prevežemo na koren druge
Ker želimo čimmanj izrojeno drevo,
vedno prevežemo manjšo množico na večjo.

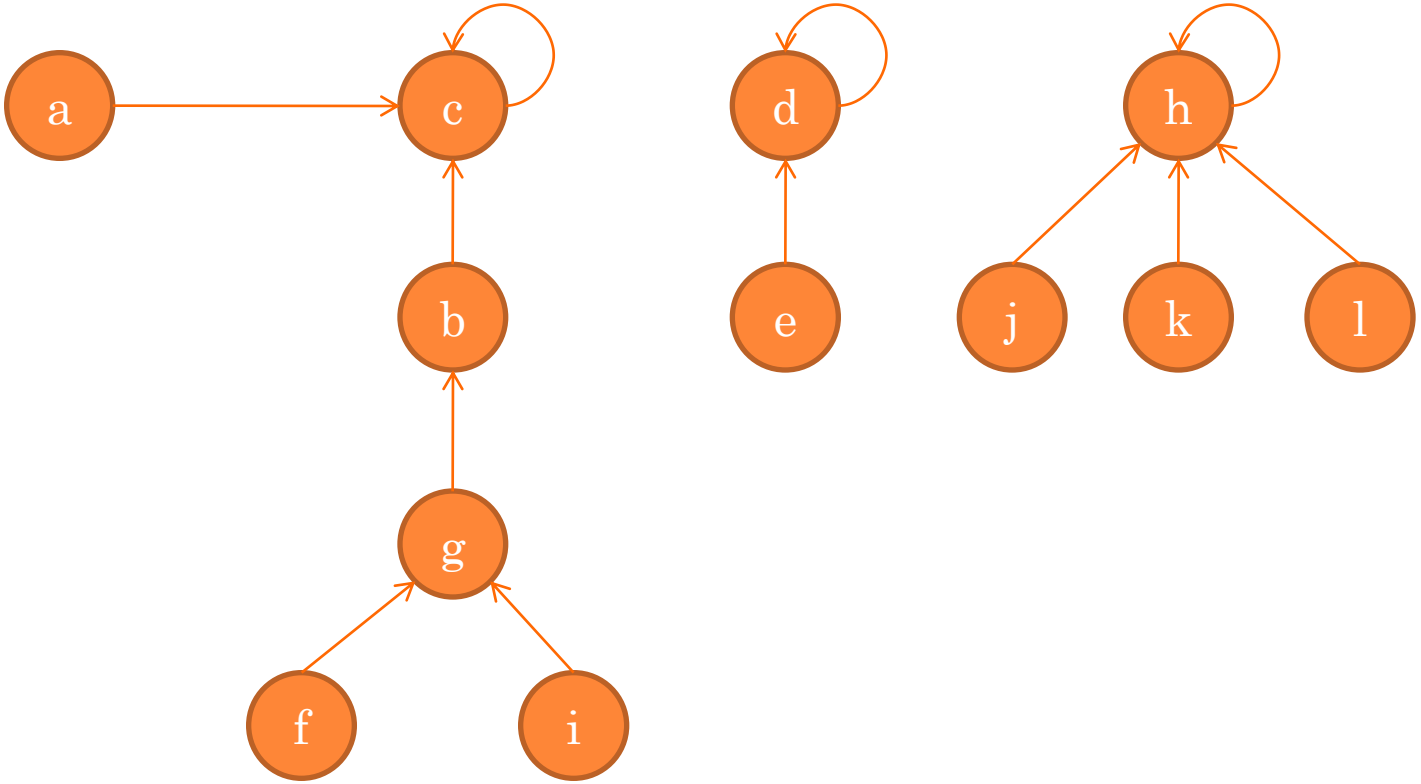
union(a,c)



# Implementacija z Gozdom

UNIJA: koren ene podmnožice prevežemo na koren druge
Ker želimo čimmanj izrojeno drevo,
vedno prevežemo manjšo množico na večjo.

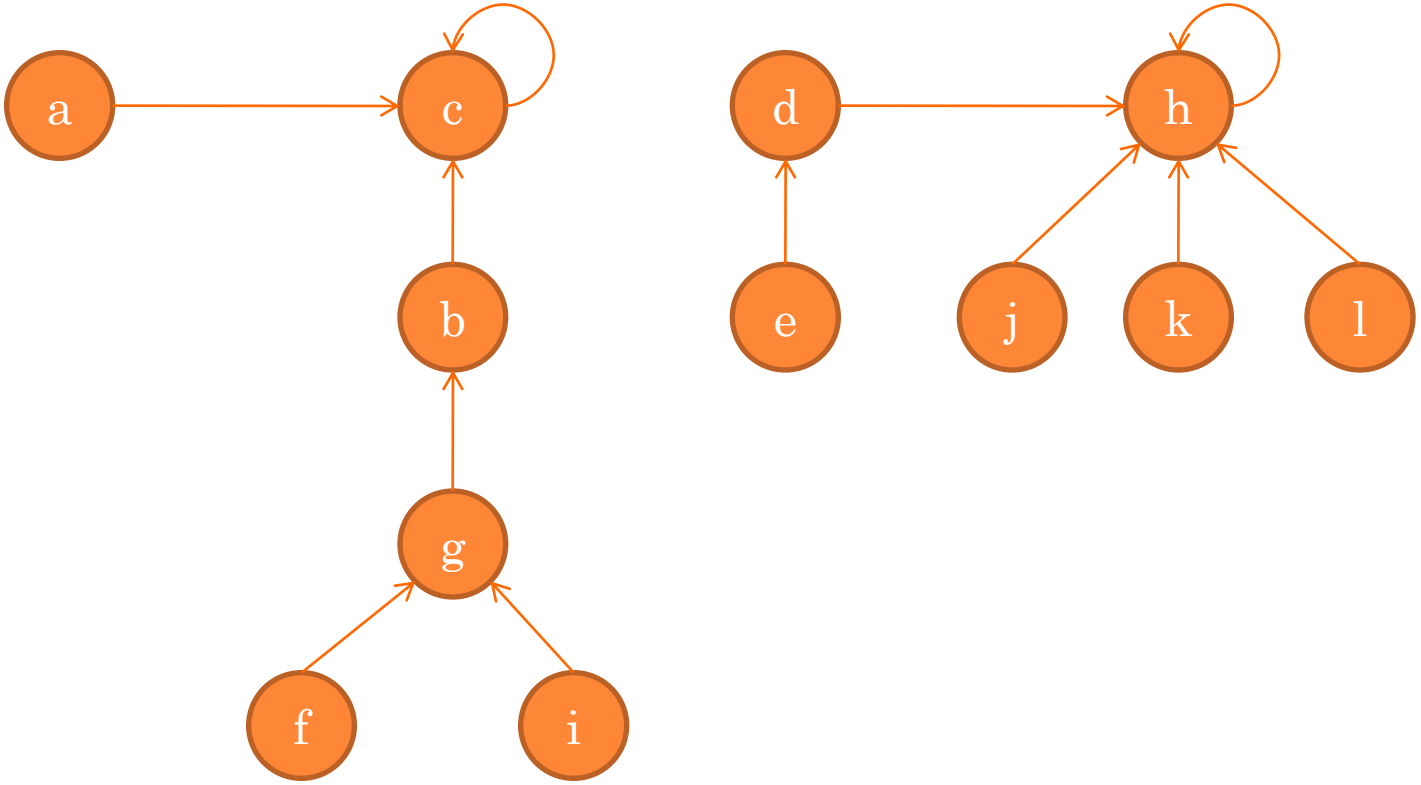
union(a,c)
union(d,h)



# Implementacija z Gozdom

UNIJA: koren ene podmnožice prevežemo na koren druge
Ker želimo čimmanj izrojeno drevo,
vedno prevežemo manjšo množico na večjo.

union(a,c)
union(d,h)



# Implementacija z Gozdom

```
public void union(DisjointSubset a1, DisjointSubset a2) {
   DisjointSubset s1 = find(a1) ;
   DisjointSubset s2 = find(a2) ;
   if (s1.noNodes >= s2.noNodes) {
      s2.parent = s1 ;
      s1.noNodes += s2.noNodes ;
   }
   else {
      s1.parent = s2 ;
      s2.noNodes += s1.noNodes ;
   }
} // union
```

Časovna zahtevnost unije: *O(1)*

# Implementacija z Gozdom

- MAKENULL: $O(1)$
- MAKESET: $O(1)$
- UNION: $O(1)$
- FIND: na dolgi rok, $m$ operacij v povprečju $O(m\ \alpha(m,n)) \approx O(m)$

  $\alpha(m,n)$ = inverzna funkcija Ackermannove funkcije = praktično konstanta

# Ackermannova funkcija

$A(1, j) = 2^j$ , $j > 0$

$A(i, 1) = A(i-1,2)$, $i > 1$

$A(i, j) = A(i-1, A(i, j-1))$, $i, j > 1$

$A(1, j) = 2^j$

$A(2, j) = A(1, A(2, j-1)) = 2^{A(2, j-1)} = 2^{2^{A(2, j-2)}} = 2^{2^{2^{A(2, j-3)}}} = 2^{2^{2^{\cdots^{2}}}}$ } $j$

$A(2, 1) = A(1, 2) = 2^2$

$A(3, 1) = A(2, 2) = 2^{2^2}$

$A(3, j) = A(2, A(3,j-1)) = 2^{2^{2^{\cdots^{2}}}}$ } $A(3,j-1)$

$A(3, 2) = 2^{2^{2^{\cdots^{2}}}}$ } $2^{2^2} = 16$ >> število atomov v vidnem vesolju