# Dizajn i analiza algoritama

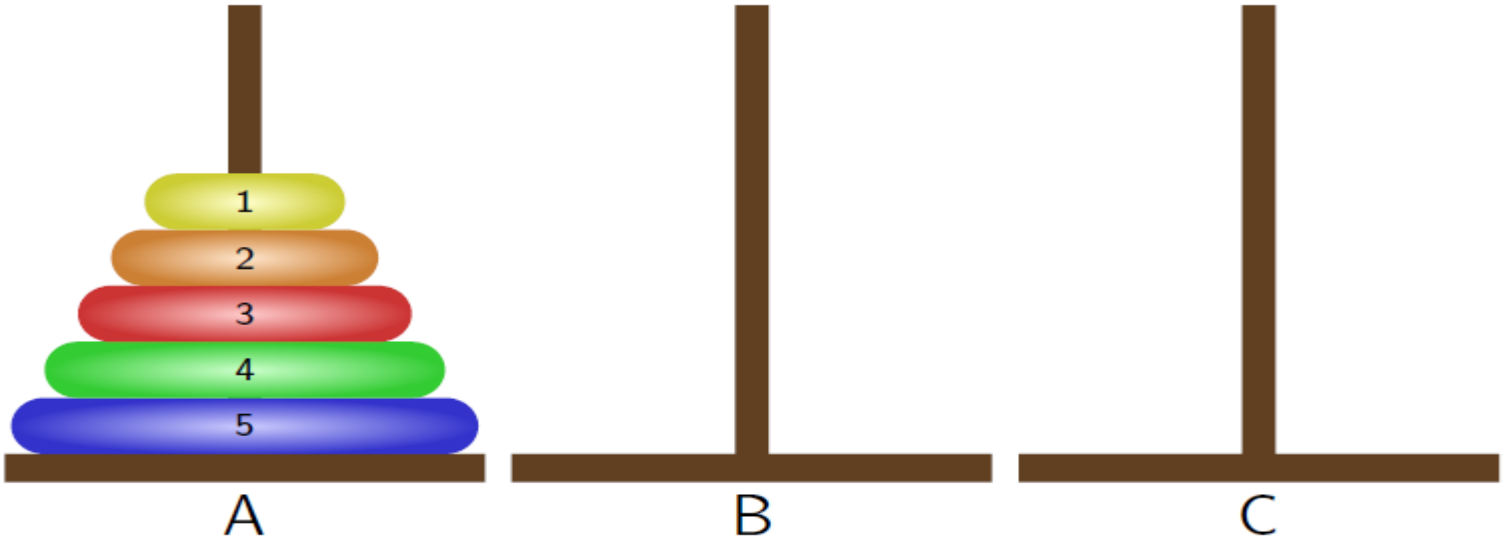
## Lekcija 8

leto 2019/2020

Prof. dr Branimir M. Trenkić

# Rekurzivni algoritmi - Primer

- ***Primer***: ***Igra Hanojske kule***
- Igra u kojoj su na početku ***n diskova*** različite veličine poređani na jednom od ***tri stuba*** u opadajućem redosledu njihovih veličina – od podnožja ka vrhu



# Rekurzivni algoritmi - Primer

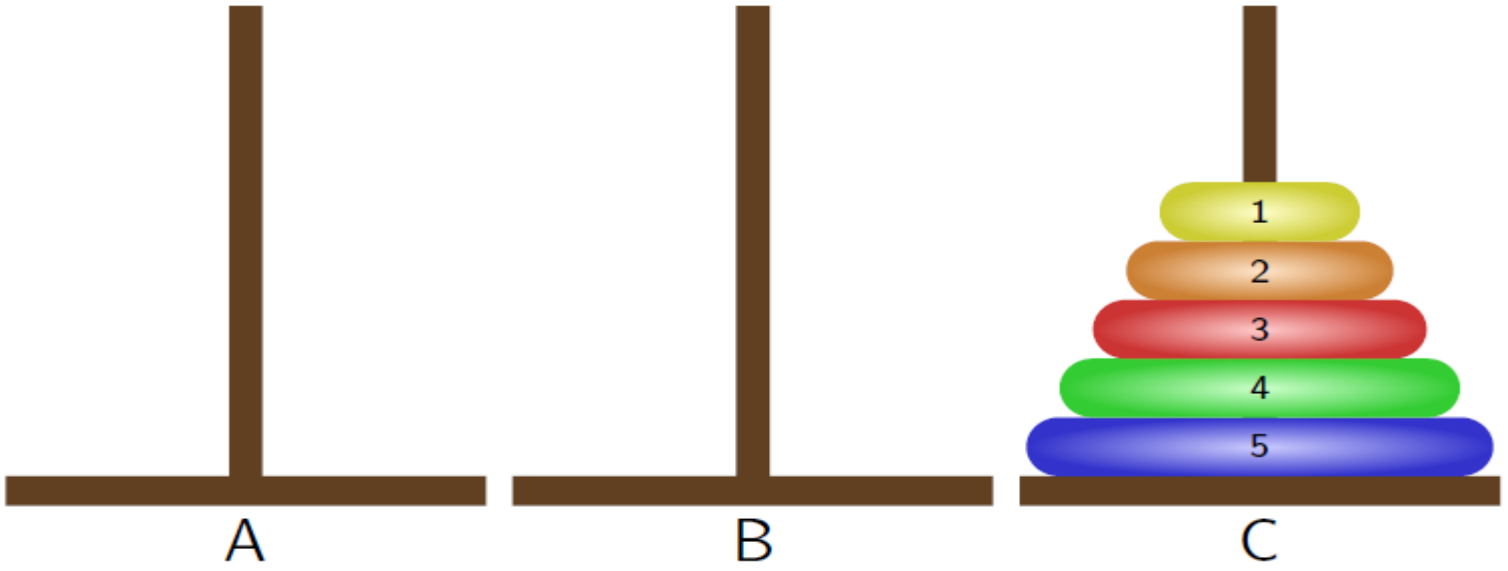
- ***Primer***: ***Igra Hanojske kule***
- Hanojske kule su zadatak (problem), koji je još davne ***1883. god***. sastavio ***francuski matematičar Edvard Lukas***

# Rekurzivni algoritmi - Primer

- **Postoji i *legenda o Hanojskim kulama*:** Legenda kaže da su postojala tri stuba na postolju hrama ***Brahma*** u Hanoju. Na levom stubu stajaše ***64 zlatna diska***, svaki različite veličine, poređani koncentrično, od najvećeg na dnu do najmanjeg na vrhu stuba. Sveštenici su imali zadatak da prebace sve diskove sa prvog stuba na drugi koristeći treći stub kada im je neophodan - ***ALI*** jedan po jedan disk i pravilo je bilo jasno, ne sme se nikad veći disk naći na manjem. Kada oni izvrše ovaj zadatak ***nastupiće kraj sveta***.

# Hanojske kule

- ***Cilj igre:*** Da se svi diskovi ***premeste*** sa početnog stuba na drugi stub – ***koristeći treći stub*** kao pomoćni. Pri tome, samo disk sa vrha jednog stuba se može premestiti na vrh drugog stuba i ***nikad*** se ***veći*** disk ne sme postaviti ***iznad manjeg***



# Hanojske kule

- ***Preciznije*** –
- Igra se sastoji od ***tri stuba***, ***A***, ***B*** i ***C***, kao i ***n*** diskova ***različitih veličina*** prečnika

$$d_1 < d_2 < \ldots\ldots < d_n$$

- ***Početno***: svi diskovi su poređani na stubu ***A*** u ***opadajućem redosledu*** – $d_n$ se nalazi na ***dnu***, a disk $d_1$ se nalazi na ***vrhu*** stuba A
- ***Cilj igre*** je premestiti sve diskove na stub ***C*** (da budu u istom redosledu) koristeći stub ***B kao pomoćni***
- Uz to, moraju se poštovati **dva pravila**!

# Hanojske kule

- ***Pravila*** u igri Hanojske kule:

1. Samo se disk ***sa vrha*** jednog stuba može premestiti ***na vrh*** drugog stuba
2. Nikad se ***veći*** disk ***ne može nalaziti iznad manjeg*** diska

- Ova pravila impliciraju:

a) Samo se ***po jedan disk*** može premeštati

b) Svi diskovi se u svakom trenutku moraju nalaziti ***samo na stubovima***

# Hanojske kule

**Demonstracija toka igre Hanojskih kula za 1, 2, 3 i 4 diska**

# Hanojske kule

- ***Rekurzivni algoritam***

- ***Ideja*** za ***rekurzivno rešenje*** Hanojskih kula sa ***n diskova***:
- Pretpostavimo da ***imamo rešenje*** ***manjeg problema*** Hanojskih kula za premeštanje ***n − 1 diska*** sa jednog stuba na drugi stub koristeći treći stub kao pomoćni
- Kako opisati ***bazni slučaj***?
- ***n = 1*** – šta je ***rešenje?***

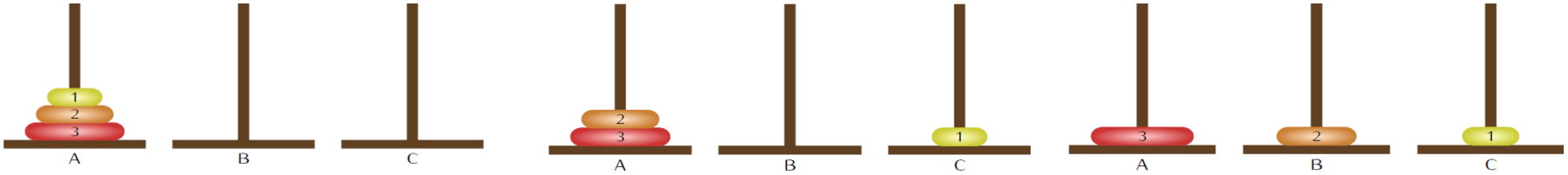
# Hanojske kule

- ***Rekurzivni algoritam***

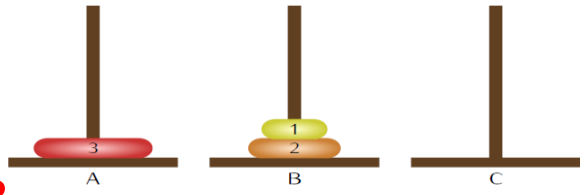
- ***Pitanje***: Kako to rešenje manjeg problema ***iskoristiti*** da bi se kreiralo rešenje za polazni problem Hanojskih kula sa ***n diskova***?

# Hanojske kule

- ***Rekurzivni algoritam***
- Primer ***n = 3*** **korak 1.**




**A-C** **A-B**




**C-B**

**Šta smo do sada uradili?**

**Premestili smo *n – 1* najmanjih diskova sa stuba A na stub B koristeći stub C kao pomoćni**

# Hanojske kule

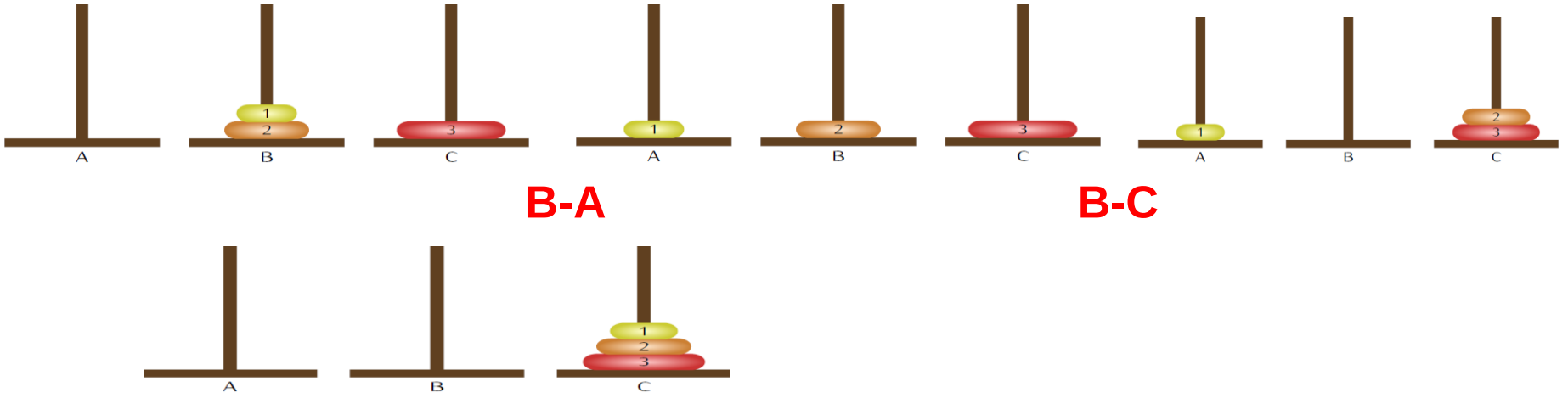
- ***Rekurzivni algoritam***
- Primer ***n = 3*** **korak 2.**


**A-C**

**Šta smo sada uradili?**
**Premestili smo *najveći disk* sa stuba A na slobodni stub C**

# Hanojske kule

- ***Rekurzivni algoritam***
- Primer ***n = 3*** **korak 3.**




**Šta smo sada uradili?**

**Premestili smo *n – 1* najmanjih diskova sa stuba B na stub C koristeći stub A kao pomoćni**

# Hanojske kule

- ***Rekurzivni algoritam***

- Polazni problem Hanojskih kula sa ***n diskova*** možemo rešiti rekurzivnim postupkom koji se sastoji iz ***tri koraka***:

1. (Rekurzivno) ***premestiti n − 1*** gornjih diskova ***sa stuba A na stub B*** koristeći stub ***C kao pomoćni***
2. Premestiti ***najveći disk sa stuba A na stub C***
3. (Rekurzivno) ***premestiti n − 1*** diskova ***sa stuba B na stub C*** koristeći stub ***A kao pomoćni***

# Hanojske kule

- ***Rekurzivni algoritam***

- Formalnije, u vidu ***rekurzivne definicije funkcije***
- ***toh(n, a, b, c)*** – opisuje korake u premeštanju ***n diskova*** ***sa*** stuba ***a*** ***na*** stub ***c***, ***koristeći*** stub ***b*** kao ***pomoćni***

$$toh(n,a,b,c)=\begin{cases} move(a,c), & n=1 \\ toh(n-1,a,c,b);move(a,c);toh(n-1,b,a,c), & n>1 \end{cases}$$

# Hanojske kule

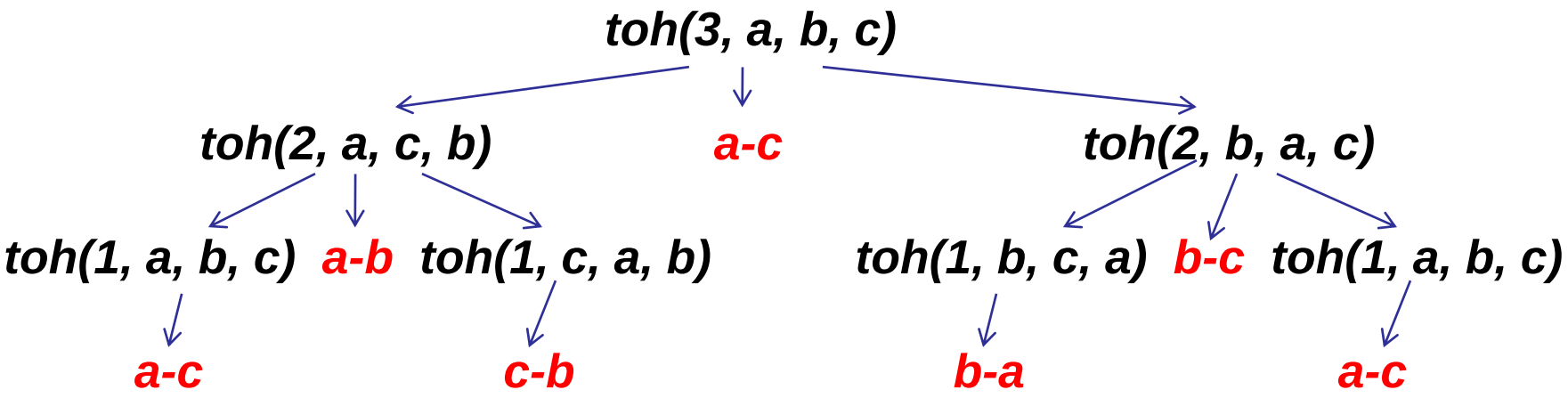
- ***Rekurzivni algoritam***

```
// Ulaz: broj diskova, početni, pomoćni i ciljni stub
// Izlaz: niz poteza za Hanojske kule sa n diskova
algorithm toh(n, a, b, c)

   if (n == 1) then  // bazni slučaj
      move(a,c);
   else              // opšti slučaj
      toh(n-1,a,c,b);
      move(a,c);
      toh(n-1,b,a,c);

   return;
```

# Hanojske kule

- ***Rekurzivni algoritam***

- Izvršavanje (***n = 3***)



# Analiza rekurzivnih algoritama

- ***Iterativni algoritmi***: ***prebrojavanje*** jediničnih ***instrukcija koje se izvode*** za vreme izvršavanja takvih algoritama
- ***Rekurzivni algoritmi***:
- ***Dinamika*** izvršavanja instrukcija ***je složenija*** – prebrojavanje komplikovano
- Vreme izvršavanja rekurzivnih algoritama se određujena drugačiji način – pomoću tzv. ***rekurentnih jednačina***

# Analiza rekurzivnih algoritama

- ***Rekurentne jednačine***
- ***Vreme izvršavanja*** rekurzivnog algoritma za ulaz ***veličine n*** ***se izražava*** pomoću vremena izvršavanja istog algoritma za ***ulaze manje od n***
- Postoji ***razvijen matematički aparat*** za rešavanje ovih jednačina
- Prema tome, na ***relativno lak način*** je moguće izvršiti procenu vremenske složenosti rekurzivnih algoritama

# Analiza rekurzivnih algoritama

- ***Primer***: ***algoritam toh*** koji rešava problem Hanojskih kula
- Neka je ***T(n)*** vreme izvršavanja ovog algoritma za ulaz veličine n (***n diskova***)
- Pomoćni algoritam ***move*** u bilo kojoj realizaciji – izvršava se ***za konstantno vreme*** – to vreme možemo uzeti kao ***jedinično vreme***

# Analiza rekurzivnih algoritama

- ***Primer***: algoritam toh koji rešava problem Hanojskih kula
- ***Analiza***:
- Za ***n = 1***, izvršava se samo pomoćni algoritam ***move***, znači ***T(1) = 1***
- Za ***n > 1***, izvršavaju se ***dva rekurzivna poziva*** istog algoritma za veličinu ulaza n – 1 i algoritam ***move***, znači ***T(n) = 2T(n-1) + 1***

$$T(n) = \begin{cases} 1, & \text{ako je } n = 1 \\ 2T(n-1)+1, & \text{ako je } n > 1. \end{cases}$$

# Analiza rekurzivnih algoritama

$$T(n) = \begin{cases} 1, & \text{ako je } n = 1 \\ 2T(n-1)+1, & \text{ako je } n > 1. \end{cases}$$

- Primer rekurentne jednačine
- ***Vrednost funkcije T(n)*** se definiše ***preko***

***a) Početnog uslova*** (u konkretnom primeru, ***T(1) = 1***)

***b) Vrednosti iste funkcije*** kada je ***argument manji od n*** (u konkretnom primeru, T(n) se predstavlja izrazom u kome učestvuje vrednost T(n-1))

# Analiza rekurzivnih algoritama

- ***Rešavanje*** rekurentnih jednačina:
- Sastoji se od ***određivanja zatvorenog izraza*** za ***T(n)*** koji će ***zavisiti samo od argumenta n*** – bez pojavljivanja vrednosti T(m) (m < n)
- Primenjuje se ***postupak višestruke zamene - opštim izrazom za T(n)*** zamenjujemo ***svaku vrednost T(m) (m < n)***
- Na taj način se ***eliminišu*** sve te vrednosti

# Analiza rekurzivnih algoritama

- ***Rešavanje*** rekurentnih jednačina:
- Na taj način se eliminišu sve te vrednosti
- ***T(n-1) zamenimo*** sa izrazom koji sadrži ***T(n-2)***
- ***T(n-2) zamenimo*** sa izrazom koji sadrži ***T(n-3)***
- ..... sve do ***T(1)*** (= 1)

# Analiza rekurzivnih algoritama

- Na taj način dobija se sledeće ***rešenje za T(n)***:

$$
\begin{aligned}
T(n) &= 2 \cdot T(n-1) + 1 \\
&= 2 \cdot \left(2 \cdot T(n-2) + 1\right) + 1 = 2^2 \cdot T(n-2) + 2 + 1 \\
&= 2^2 \cdot \left(2 \cdot T(n-3) + 1\right) + 2 + 1 = 2^3 \cdot T(n-3) + 2^2 + 2 + 1 \\
&\vdots \\
&= 2^{n-1} \cdot T(1) + 2^{n-2} + 2^{n-3} + \cdots + 2^2 + 2 + 1 \\
&= 2^{n-1} \cdot 1 + \sum_{i=0}^{n-2} 2^i = 2^{n-1} + \frac{2^{n-1} - 1}{2 - 1} = 2 \cdot 2^{n-1} - 1 \\
&= 2^n - 1.
\end{aligned}
$$

# Analiza rekurzivnih algoritama

- Dakle, možemo zaključiti da je $T(n) = 2^n - 1$
- ***T(n)*** je zapravo ***broj poteza u rešavanju problema*** Hanojskih kula za n diskova
- Izvršavanje pomoćnog algoritma ***move predstavlja jedan potez u rešavanju***
- *Primer*

  *move* = 1 sek.

  *T(n)* = 585 milijardi godina (n = 64)