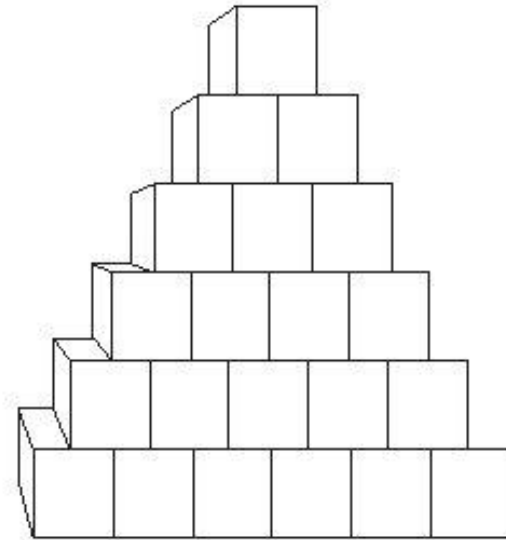


Dinamičko programiranje

Dinamičko programiranje

- Optimizacija (min, max)
- Potproblemi
- Gledanje unazad
 - Ne gledamo šta ćemo sledeće da uzmemo
 - Već šta smo uzeli prethodno



Rešavanje problema - koraci

1. Struktura optimalnog rešenja
2. Rekurzivna definicija
3. Izračunavanje rešenja „odozdo na gore“
4. Rekonstrukcija rešenja

Zadatak 1

- Dat je niz A od N celih brojeva
- Odrediti podniz niza A čiji zbir elemenata je maksimalan, a u kome nema susednih elemenata niza A .
- Smatrati da prazan niz ima zbir elemenata 0.
- Naći sve moguće kombinacije – eksponencijalna kompleksnost

Korak 1 – struktura rešenja

- Potproblemi – različite dužine niza
- $a_1 \ a_2 \ \dots \ a_{i-2} \ a_{i-1} \ a_i \ \dots \ a_n$
- 2 slučaja
 - a_i deo rešenja – dodati na optimum do pozicije $i - 2$
 - a_i nije deo rešenja – zadržati optimum do pozicije $i - 1$
- Biramo veći rezultat
- Koristimo rešenja prethodnih potproblema da rešimo trenutni potproblem.

Korak 2 - Rekurzija

- $opt(i) = \max \left(a_i + opt(i - 2), \right. \\ \left. opt(i - 1) \right)$
- $opt(1) = \max(0, a_1)$
- $opt(0) = 0$
- Problem – opet eksponencijalna kompleksnost
- Više puta se rešavaju isti potproblemi

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

A	3	5	2	4	3	1	6
opt							
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije
- $opt[0]=0$

A	3	5	2	4	3	1	6
opt	0						
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije
- $\text{opt}[0]=0$
 $\text{opt}[1]=\max(0, a[1])$

A	3	5	2	4	3	1	6
opt	0	3					
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6
opt	0	3	5				
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6
opt	0	3	5	5			
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6
opt	0	3	5	5	9		
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6	
opt	0	3	5	5	9	9		
	0	1	2	3	4	5	6	7

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6
opt	0	3	5	5	9	9	10
	0	1	2	3	4	5	6

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$

$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6	
opt	0	3	5	5	9	9	10	15
	0	1	2	3	4	5	6	7

Korak 3 – odozdo na gore (bottom-up)

- Bolje rešenje – suprotni pravac od rekurzije

- $\text{opt}[0]=0$

$\text{opt}[1]=\max(0, a[1])$

for $i=2, n$


$\text{opt}[i]=\max(a[i]+\text{opt}[i-2], \text{opt}[i-1])$

A	3	5	2	4	3	1	6	
opt	0	3	5	5	9	9	10	15
	0	1	2	3	4	5	6	7

← optimalno rešenje,
maksimalni zbir


Korak 4 – rekonstrukcija rešenja

- Koji elementi učestvuju u rešenju?
- Oni za koje važi $opt_i \neq opt_{i-1}$
- Krećemo od kraja niza

A							
	3	5	2	4	3	1	6
opt	0	3	5	5	9	9	10
	0	1	2	3	4	5	6

Korak 4 – rekonstrukcija rešenja

- Koji elementi učestvuju u rešenju?
- Oni za koje važi $opt_i \neq opt_{i-1}$
- Krećemo od kraja niza




A	3	5	2	4	3	1	6	
opt	0	3	5	5	9	9	10	15
	0	1	2	3	4	5	6	7

Korak 4 – rekonstrukcija rešenja

- Koji elementi učestvuju u rešenju?
- Oni za koje važi $opt_i \neq opt_{i-1}$
- Krećemo od kraja niza

Korak 4 – rekonstrukcija rešenja


- Koji elementi učestvuju u rešenju?
- Oni za koje važi $opt_i \neq opt_{i-1}$
- Krećemo od kraja niza



A	3	5	2	4	3	1	6	
opt	0	3	5	5	9	9	10	15
	0	1	2	3	4	5	6	7

Korak 4 – rekonstrukcija rešenja

- Koji elementi učestvuju u rešenju?
- Oni za koje važi $opt_i \neq opt_{i-1}$
- Krećemo od kraja niza



A	3	5	2	4	3	1	6
---	---	---	---	---	---	---	---

opt	0	3	5	5	9	9	10	15
	0	1	2	3	4	5	6	7

Traženi podniz:

5, 4, 6

Test primer 2

A	3	-5	2	4	-3	-1	6	
opt								
	0	1	2	3	4	5	6	7

Test primer 2

A	3	-5	2	4	-3	-1	6	
opt	0	3	3	5	7	7	7	13
	0	1	2	3	4	5	6	7

Traženi podniz:

3, 4, 6

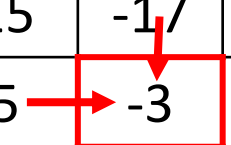
Zadatak 2

- Data je matrica A sa M vrsta i N kolona, popunjena celim brojevima.
- Sa svakog polja je dozvoljeno preći samo na polje ispod ili na polje desno od tog polja.
- Potrebno je izabrati put od gornjeg levog do donjeg desnog polja tako da zbir brojeva u poljima preko kojih se ide bude maksimalan.
- Ispisati vrednost optimalnog puta, a zatim i put kao niz koordinata polja preko kojih se ide.

Korak 1 – struktura rešenja

- Potproblemi – zbir u različitim poljima matrice

2	15	-17	16
14	5	-3	-14
-19	-12	5	-16



- 2 slučaja
 - Na polje a_{ij} se stiglo odozgo – dodati na optimum na polju $i - 1, j$
 - Na polje a_{ij} se stiglo sleva – dodati na optimum na polju $i, j - 1$

Korak 2 - Rekurzija

- $opt(i, j) = \max \left(\begin{array}{l} a_{ij} + opt(i - 1, j), \\ a_{ij} + opt(i, j - 1) \end{array} \right)$
- $opt(1, j) = \sum_{k=1}^j a_{1k}$
- $opt(j, 1) = \sum_{k=1}^j a_{k1}$

Korak 3 – bottom-up

A

2	15	-17	16
14	5	-3	-14
-19	-12	5	-16

opt

2	17	0	16
16	22	19	5
-3	10	24	8

Korak 4 – Rekonstrukcija

A

2	15	-17	16
14	5	-3	-14
-19	-12	5	-16

opt

2	17	0	16
16	22	19	5
-3	10	24	8

- Krenemo od donjeg desnog ugla
- Idemo gore ili desno, zavisno gde je veći opt

Zadatak 3 (za domaći)

- Neka N ljudi čeka u redu da kupi karte za predstavu, pri čemu je t_i vreme koje je i -tom kupcu potrebno da kupi kartu.
- Ako se po dvoje suseda u redu udruži da kupi karte, npr. k -ti i $k + 1$ -vi kupac, onda je vreme potrebno da oni kupe karte p_k , $k = 1..n - 1$.
- Udruživanje može da ubrza kupovinu, a i ne mora.
- Za dato n i nizove t i p odrediti takav način udruživanja da ukupno vreme potrebno da svih n kupaca kupi kartu bude minimalno.