



## Naloga 2

### Naloga 2 - množenja

Naloga je sestavljena iz dveh sklopov. V prvem sklopu boste implementirali 3 algoritme za množenje velikih števil. V drugem sklopu pa 3 algoritme za množenje matrik. Prvi argument programu bo ločil ta dva sklopa:

- `num` - množenje velikih celih števil;
- `mat` - množenje matrik

#### Množenje števil

Implementirajte naslednje algoritme (drugi argument programa bo izbral ustrezni algoritem):

- `os` - osnovnošolsko množenje;
- `dv` - naivni deli in vladaj;
- `ka` - Karacubov algoritem.

**Predstavitev števil.** Števila predstavite v ustrezno veliki tabeli v desetiškem zapisu - torej eno polje tabele naj hrani eno števk (0-9) tega števila. Vedno delajte z optimalno dolžino števila - torej število naj nima nikoli odvečnih ničel na začetku.

**Branje vhoda.** Pri vsakem testu bosta na vhodu podani dve števili (vsako v svoji vrstici), ki ju morate zmnožiti z enim izmed algoritmov. Števili bosta vedno v desetiškem zapisu in bosta pozitivni. Število števk v enem številu bo  $\leq 10^3$ .

**Sled algoritma.** Format sledi algoritma je odvisen od vsakega algoritma posebej in bo podan v nadaljevanju.

#### Osnovnošolsko množenje

Kot prvi algoritem implementirajte osnovnošolsko množenje, torej algoritem, ki se ga vsi naučimo v osnovni šoli. Prvo število zmnožimo z vsemi števki drugega, potem pa seštejemo vsa dobljena števila - seveda ustrezno zamaknjena.

**Format sledi:**

Sled izpišite v podobni obliki, kot je navada pri računanju "na roko". Vsako števko prvega števila pomnožite z vsako števko drugega števila.

Po vsakem množenju z eno števko izpišite zmnožek, vsak v svoji vrstici.

Nato izpišite vrstico, ki ločuje delne rezultate in skupno vsoto (to naj bodo znaki "-", ki naj jih bo toliko, kot je dolžina rezultata). V zadnji vrstici pa izpišite rezultat.

**Primer izvajanja:**

Standardni vhod:

```
456
103
```

Izpis sledi:

```
java Naloga2 num os
456
0
1368
-----
46968
```

**Naivni deli in vladaj algoritem**

Implementirajte naivni algoritem deli in vladaj, kot smo ga spoznali na vajah. Ker je možnih implementacij zelo veliko, se obvezno držite spodnjih nasvetov.

Vedno predpostavite, da sta obe števili enake dolžine in da je ta dolžina soda. Če temu ni tako, si lahko predstavljate, da je število spredaj podaljšamo z navideznimi ničlami.

Denimo, da vhodni števili razstavimo na dve enako dolgi  $a = a_1a_0$  in  $b = b_1b_0$ , produkt teh dveh števil pa izračunate kot  $ab = a_1b_110^n + (a_0b_1 + a_1b_0)10^{\frac{n}{2}} + a_0b_0$ .

Rekurzivno kličite produkte v tem zaporedju:  $a_0b_0$  nato  $a_0b_1$ ,  $a_1b_0$  in nazadnje  $a_1b_1$ .

Ustavitvena pogoja rekurzije naj bosta:

1. Če je katero izmed števil 0, potem vrnete kar 0.
2. Če je katero izmed števil samo ena številka, potem kar pomnožimo drugo število s to številko (enostavno linearno množenje) .

**Format sledi:**

Vsakič, ko je poklicana metoda za množenje, izpišite v eni vrstici dve števili, ki ju množimo, takoj ko pa dobite rezultat množenja (torej se vrnete iz rekurzije) izpišite rezultat množenja.

**Primer izvajanja:**

Standardni vhod:

```
123
456
```

Izpis sledi:

```
java Naloga2 num dv
123 456
23 56
3 6
18
3 5
15
2 6
12
2 5
10
1288
23 4
92
1 56
56
1 4
4
56088
```

### Karacubov algoritem

Kot zadnji algoritem implementirajte Karacubov algoritem, ki smo ga spoznali na vajah.

Pri implementaciji se držite istih predpostavk kot pri enostavnem deli in vladaj algoritmu (predvsem glede dolžine števil in ustavitvenih pogojev).

Produkt dveh števil je definiran kot

$$ab = a_1b_110^n + (((a_0 + a_1)(b_0 + b_1) - a_1b_1 - a_0b_0)10^{\frac{n}{2}} + a_0b_0).$$

Rekurzivno kličete produkte v tem zaporedju:  $a_0b_0$ ,  $a_1b_1$  in nazadnje  $(a_0 + a_1)(b_0 + b_1)$ .

#### Format sledi:

Sled naj bo podobna (enaka) sledi pri enostavnem algoritmu deli in vladaj, ob vsakem rekurzivnem klicu metode izpišite tisti dve števili, ki ju množimo. Ko pa se vrnemo iz rekurzivnega klica, takoj izpišemo dobljeni zmnožek.

#### Primer izvajanja:

Standardni vhod:

```
1234
12
```

Izpis sledi:

```
java Naloga2 num ka
1234 12
34 12
4 2
8
3 1
3
7 3
21
408
12 0
0
46 12
6 2
12
4 1
4
10 3
30
552
14808
```

## Množenje matrik

Implementirajte naslednje algoritme (drugi argument programa bo izbral ustrezni algoritem):

- os - enostavno (osnovno) množenje;
- dv - naivni deli in vladaj;
- st - Strassenov algoritem.

## Format vhoda

Na standardnem vhodu boste dobili dve matriki, ki ju morate zmnožiti. Za vsako matriko bosta najprej podani dve dimenziji:  $n$  - število vrstic matrike in  $m$  - število stolpcev matrike. Sledi  $n*m$  celih števil (matrika podana po vrsticah)

### Primer vhoda:

```
2 3
1 2 3
4 5 6
3 2
1 1
1 1
1 1
```

## Enostavno množenje

Prvi algoritem je zgolj implementacija po definiciji. Kot sled izpišite samo rezultat množenja. V prvi vrstici izpišite dimenzijo matrike (v obliki DIMS: NxM), potem pa matriko po vrsticah, števila v vrstici pa ločena s presledkom.

### Primer izvajanja

java Naloga2 mat os

```
2 3
1 1 1
2 2 2
3 2
1 1
2 2
3 3
```

Na standardni izhod dobimo izpis:

```
DIMS: 2x2
6 6
12 12
```

### Naivni deli in vladaj

Pri enostavnem deli in vladaj algoritmu bomo predpostavili, da sta obe matriki enako veliki in kvadratni. Poleg tega naj bo njuna dimenzija enaka  $2^k$ . Če na vhodu dobimo matrike, ki temu pogoju ne zadoščajo, potem jih dopolnimo (do dimenzije, ki je najbližja potenca števila 2) z vrsticami in stolpci, ki vsebujejo same ničle. Taki dve matriki nato razdelimo na štiri enake dele in končni rezultat izrazimo na sledeči način:

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} (A_{11}B_{11} + A_{12}B_{21}) & (A_{11}B_{12} + A_{12}B_{22}) \\ (A_{21}B_{11} + A_{22}B_{21}) & (A_{21}B_{12} + A_{22}B_{22}) \end{bmatrix}$$

Vsaakega od osmih produktov nato izračunamo rekurzivno, dokler nimamo opravka z matrikama dimenzije 1x1, kjer produkt trivialno izračunamo. Seveda se pri rekurziji ustavimo lahko že prej, zato naj ima vaš program še en dodaten parameter, ki pove pri kako velikih matrikah naj se rekurzija ustavi (privzeta vrednost naj bo 1).

Produkte kličite rekurzivno v tem zaporedju:

$A_{11}B_{11}$ ,  $A_{12}B_{21}$ ,  $A_{11}B_{12}$ ,  $A_{12}B_{22}$ ,  $A_{21}B_{11}$ ,  $A_{22}B_{21}$ ,  $A_{21}B_{12}$ ,  $A_{22}B_{22}$

### Format sledi

Takoj, ko imate enega od teh delnih produktov izračunanih, izpišite vsoto vseh elementov v tem delnem produktu.

Nazadnje izpišite še rezultat množenja v formatu kot je bil definiran pri enostavnem množenju.

### Primer izvajanja

java Naloga2 mat dv 2

```

2 3
1 1 1
2 2 2
3 2
1 1
2 2
3 3

```

Na standardni izhod dobimo:

```

18
18
0
0
0
0
0
0
0
DIMS: 4x4
6 6 0 0
12 12 0 0
0 0 0 0
0 0 0 0

```

## Strassenov algoritem

Strassenov algoritem tudi deluje po principu deli in vladaj, zato se držite enakega napotka kot zgoraj, torej dimenzije obeh matrik vedno dopolnite, da dobite dve kvadratni matriki, dimenzija pa naj bo najbližja (večja) potenca števila 2.

Pri tem algoritmu imamo 7 produktov polovične velikosti, ki jih lahko navedemo kot:

$$P_1 = A_{11}(B_{12} - B_{22})$$

$$P_2 = (A_{11} + A_{12})B_{22}$$

$$P_3 = (A_{21} + A_{22})B_{11}$$

$$P_4 = A_{22}(B_{21} - B_{11})$$

$$P_5 = (A_{11} + A_{22})(B_{11} + B_{22})$$

$$P_6 = (A_{12} - A_{22})(B_{21} + B_{22})$$

$$P_7 = (A_{11} - A_{21})(B_{11} + B_{12})$$

Iz teh sedmih produktov skonstruiramo produktno matriko (rezultat množenja matrik A in B):

$$\begin{bmatrix} (P_5 + P_4 - P_2 + P_6) & (P_1 + P_2) \\ (P_3 + P_4) & (P_1 + P_5 - P_3 - P_7) \end{bmatrix}$$

Delne produkte izračunajte rekurzivno v zaporedju  $P_1, P_2, \dots, P_7$ .

**Format sledi**

Format izpisa je zelo podoben kot pri naivnem deli in vladaj algoritmu. Takoj, ko imate naračunan produkt  $P_i$ , izpišite vsoto vseh elementov v tej matriki na standardni izhod.

Nazadnje izpišite še rezultat množenja, kot je bilo to pri naivnem deli in vladaj algoritmu.

### Primer izvajanja

java Naloga2 mat st 2

```
2 3
1 1 1
2 2 2
3 2
1 1
2 2
3 3
```

Na standardni izhod dobimo:

```
0
0
0
0
18
18
18
DIMS: 4x4
6 6 0 0
12 12 0 0
0 0 0 0
0 0 0 0
```

Če pa program izvedemo s parametri:

java Naloga2 mat st

potem pa dobimo na standardni izhod:

0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
-1  
4  
4  
2  
9  
-4  
-2  
18  
3  
0  
6  
0  
3  
0  
-6



```

18
-1
4
4
2
9
-4
-2
18
DIMS: 4x4
6 6 0 0
12 12 0 0
0 0 0 0
0 0 0 0

```

## Status oddaje naloge

Status oddaje naloge	Pri tej nalogi vam ni treba oddati ničesar.
	Pri tej nalogi ne morete oddati prispevkov.
Stanje ocen	Neocenjeno
Rok za oddajo	nedelja, 6. maj 2018, 23:55
Preostali čas	Rok za oddajo naloge je potekel
Zadnja sprememba	-
Komentar oddaje	► Komentarji (0)

## Odziv

Ocena	8,00 / 8,00
Ocenjeno v	ponedeljek, 4. junij 2018, 11:45

### NAVIGACIJA



#### Pregledna plošča

- Prva stran
- Strani spletnega mesta











## Trenutni predmet

aps2uni

Sodelujoči

Priznanja

Splošno

 O predmetu Potek predmeta Samostojno delo Viri in literatura Obvestila Razprave Opravljanje domačih nalog Naloga 0 Naloga 1 **Naloga 2** Naloga 3 Skupaj izzivi Dodatne točke teorija3-rok

5. marec - 11. marec

12. marec - 18. marec

19. marec - 25. marec

26. marec - 1. april

2. april - 8. april

9. april - 15. april

16. april - 22. april

23. april - 29. april

7. maj - 13. maj

14. maj - 20. maj

21. maj - 27. maj

28. maj - 3. junij

Moji predmeti

Predmeti

## NASTAVITVE



Skrbništvo predmeta