KĀRTOŠANA UN MEKLĒŠANA III

42.nodarbība

«SKALDI UN VALDI» VĒSTURĒ UN POLITIKĀ

- Lielu spēku sadala vairākos mazākos, katrs no kuriem ir vājāks par pretinieku, kas izmanto šo stratēģiju.
- Jūlijs Cēzars
- Bonaparts Napoleons
- Daudzas partijas ar līdzīgiem politiskajiem principiem (mēs būsim labāki par iepriekšējiem..).

«SKALDI UN VALDI» KĀ METODE PROBLĒMU RISINĀŠANAI

- Princips – lielu problēmu sadalīt mazākās problēmās, kuras mēs protam atrisināt.
- Piemēram, mēs protam iesmelt krūzē ūdeni un to izliet.
- Iespējamais risinājums, lai iztukšotu mucu: atkārtot iepriekšējo darbību līdz muca ir tukša.

«SKALDI UN VALDI» KĀ KĀRTOŠANAS METODE

Vispirms pārliecināsimies, ka ir viegli no diviem jau sakārtotiem masīviem iegūt vienu sakārtotu masīvu.

Sapludināšanas algoritms:

Aija Dace Toms Vilnis

Elza Ieva Ivo Santa Svens

Aija Dace Elza Ieva Ivo Santa Svens Toms Vilnis

«SKALDI UN VALDI» KĀRTOŠANAS ALGORITMS

Algoritms SAKĀRTOT(elementus no sākuma līdz beigām)

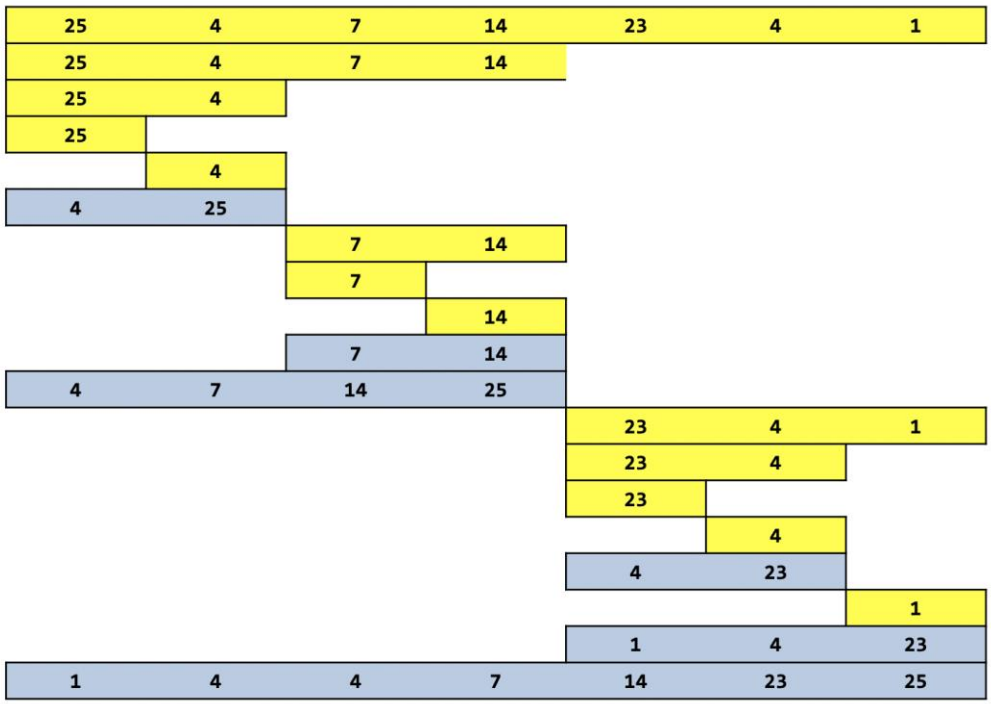
Ja sākums<beigām tad:

SAKĀRTOT(elementus no sākuma līdz vidum)

SAKĀRTOT (elementus no vidus+1 līdz beigām)

Sapludināt sakārtotās daļas

PIEMĒRS – «SKALDI UN VALDI» KĀRTOŠAS ALGORITMS DARBĪBĀ



«SKALDI UN VALDI» ALGORITMA NOVĒRTĒJUMS

- Priekšrocība - $O(n)=n*\log_2 n$
- Trūkumi:
  - Neinformēta metode – jau sakārtotu masīvu kārtos tikpat ilgi kā nesakārtotu.
  - Vajadzīga papildus vieta O(n) atmiņā, kur sapludināt jau sakārtotās daļas.

- Papildus vieta atmiņā vajadzīga arī rekursijas realizēšanai, taču to var nedaudz ietaupīt, realizēt algoritmu bez rekursijas.

KAD UN KĀPĒC VAJAG TAUPĪT DATORA RESURSUS?

Ja ir lieli datu apjomi.

Piena ceļa galaktikā atrodas apmēram 100 miljardi zvaigžņu. Ja gribam saglabāt par katru zvaigzni kaut vai tikai 50B informācijas, tad viss kopā vismaz 5000 000 000 000B=5000GB. Operatīvajā atmiņā to ielasīt nevarēs!

Ja ir jāveic daudz aprēķinu.

Latvijā ir ap 2 00 000 skolēnu. Ja katram ir nepieciešams aprēķināt vidējo atzīmi katrā mācību priekšmetā, tad var sanākt veikt pat 2 000 000 summēšanu.

OLIMPIĀŽU UZDEVUMA PIEMĒRS NO OLIMPS.LIO.LV

labi	**Labi sakārtotie skaitļi**	1 sek.

Uzdevums

Par *labi sakārtotu* sauksim tādu **n**-ciparu nenegatīvu veselu skaitli **a** = $\mathbf{a_n a_{n-1} \ldots a_2 a_1}$, kura cipariem ir spēkā sekojoša sakarība: $\mathbf{a_n} >= \mathbf{a_{n-1}} >= \ldots >= \mathbf{a_2} >= \mathbf{a_1}$. Dotai **n** vērtībai noteikt, cik pavisam ir **n** ciparu labi sakārtoti skaitļi.

Ievaddati

Teksta faila *labi.in* pirmajā rindā dota naturāla skaitļa **n** (0<**n**<41) vērtība.

Izvaddati

Teksta faila *labi.out* pirmajā rindā jāizvada viens naturāls skaitlis - **n** ciparu labi sakārtoto skaitļu skaits.

Piemērs

labi.in	labi.out
2	54

Piezīme: Šie skaitļi ir: 10,11,20,21,22,30,31,32,33,40,41,42,43,44,

50,51,52,53,54,55,60,61,62,63,64,65,66,70,71,72,73,74,75,76,

77,80,81,82,83,84,85,86,87,88,90,91,92,93,94,95,96,97,98,99.

UZDEVUMA «LABI SAKĀRTOTI SKAITĻI» ANALĪZE

Nosacījums 0<**n**<41

n=1

visi skaitļi ir labi sakārti, atbilde ir 10

n=2

der 10, 11, 20, 21, 22, 30, 31, 32, 32, 33 ..

derīgo skaitļu skaits ir 2+3+4+5+6+7+8+9+10=54

UZDEVUMA «LABI SAKĀRTOTI SKAITĻI» NOVĒRTĒJUMS

n=3

Cik vispār ir trīs ciparu skaitļu? 900

Derīgo skaitļu skaits<900

n=4

Cik vispār ir četrciparu skaitļu? 9000=10000-1000

Derīgo skaitļu skaits<9000

n=40

Cik tādu skaitļu ir? 10..000-10..00=?

Derīgo skaitļu skaits<

900000000000000000000000000000000000000

PROCESORA ĀTRDARBĪBA

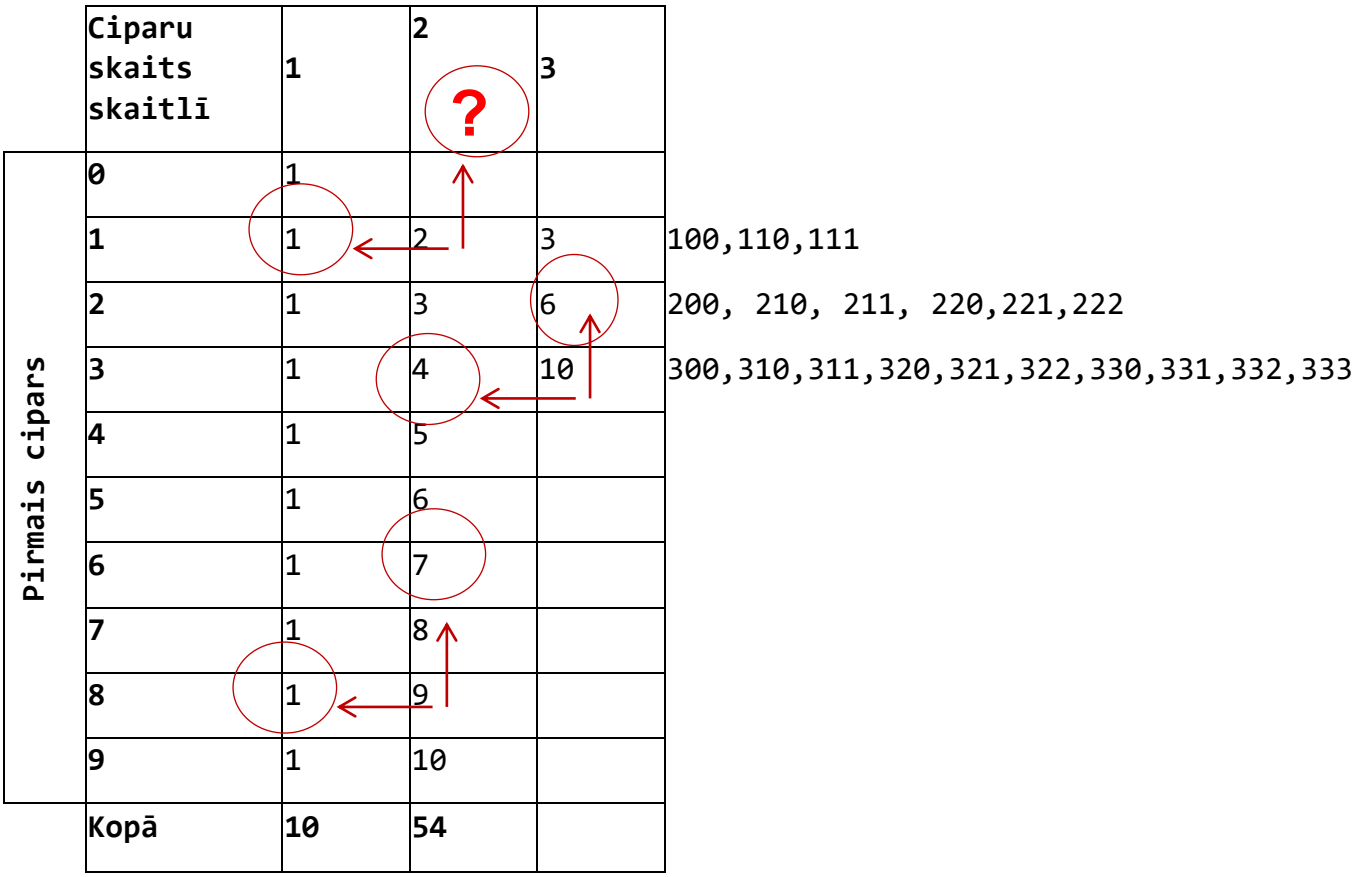
- 1.8GHz
- 1GHz ≅ 1 000 000 000 darbības 1 sekundē
- Cik sekundes vajadzēs, lai pārbaudītu visus 40-ciparu skaitļus?

900000000000000000000000000000s ☹

MĒĢINĀSIM IZRĒĶINĀT..

	Ciparu skaits skaitlī	1	2	3
Pirmais cipars	0	1		
	1	1	2	
	2	1	3	
	3	1	4	
	4	1	5	
	5	1	6	
	6	1	7	
	7	1	8	
	8	1	9	
	9	1	10	
	Kopā	10	54	

VAI VARI IERAUDZĪT LIKUMSAKARĪBU?



FORMULA JEB PROCESORA RESURSU TAUPĪŠANA

skaits	Ciparu skaits skaitlī	1	2	3
Pirmais cipars	0	1	1	1
	1	1	2	3
	2	1	3	6
	3	1	4	10
	4	1	5	15
	5	1	6	21
	6	1	7	28
	7	1	8	36
	8	1	9	45
	9	1	10	55
	Kopā	10	54	219

00 000

$skaits[i][n] = skaits[i][n-1] + skaits[i-1][n]$

Kopā $n-ciparu$ labi sakārtoti skaitļi

ja n=1, tad 10

ja n>1, tad $\sum_{j=1}^{9} skaits[j][n]$

RESURSU PATĒRIŅŠ – VAI VARAM VĒL KAUT KO IETAUPĪT?

```
public class LabiSakartotiSkaitli{
  public static void main(String[] args){
    int n=40;
    int[][] skaits=new int[10][40];
    //iedodam sākuma vērtības
    for(int i=0;i<10;i++)skaits[i][0]=1;
    //aizpildām tabulu
    for(int j=1;j<n;j++){
        skaits[0][j]=1;
        for(int i=1;i<10;i++){
            skaits[i][j]=skaits[i][j-1]+skaits[i-1][j];
        }

    }
    //apreekinam peedeejas kolonnas summu
    int sum=0;
    for(int i=1;i<10;i++){
        sum=sum+skaits[i][n-1];
    }
    //izvadām rezultātu
    if(n==1)System.out.println(10);
    else System.out.println(sum);
  }
}
```

RAM 4*10*40 = **1600B**
Ātrdarbība:
O(10)+O(n-1)+O((n-1)*9)+O(9)
≅ **O(10*n)**

RAM JEB OPERATĪVĀS ATMIŅAS RESURSU TAUPĪŠANA

```
public class LabiSakartotiSkaitli{
  public static void main(String[] args){
    int n=40;
    int[][] skaits=new int[10][2];
    //iedodam sākuma vērtības
    for(int i=0;i<10;i++)skaits[i][0]=1;
    //aizpildām tabulu
    for(int j=1;j<n;j++){
        skaits[0][1]=1;
        for(int i=1;i<10;i++){
            skaits[i][1]=skaits[i][0]+skaits[i-1][1];
        }
        for(int i=1;i<10;i++){
            skaits[i][0]=skaits[i][1];
        }
    }
    //apreekinam peedeejas kolonnas summu
    int sum=0;
    for(int i=1;i<10;i++){
        sum=sum+skaits[i][1];
    }
    //izvadām rezultātu
    if(n==1)System.out.println(10);
    else System.out.println(sum);
  }
}
```

RAM 4*10*2 = **80B**
Ātrdarbība:
O(10)+O(n-1)+O((n-1)*18)
≅ **O(20*n)**

	Ciparu skaits skaitlī	1	2
Pirmais cipars	0	2	1
	1	3	2
	2	4	3
	3	5	4
	4	6	5
	5	7	6
	6	8	7
	7	9	8
	8	10	9
	9	1	10
	Kopā	10	54

KĀ NOTEIKT PROGRAMMAS IZPILDES LAIKU?

```
1  public class LabiSakartotiSkaitli{
2    public static void main(String[] args){
3      int n=1000000000;
4      int skaits=0;
5      //nolasa un saglabâ sâkuma laiku
6      long ms=System.currentTimeMillis();
7      //pârbauda visus iespeejamos n-ciparu skaitlus
8      for(int i=n/10;i<n;i++){
9
10     }
11     //izvada iegûto skaitu
12     System.out.println(skaits);
13     //izvada starpîbu starp sâkuma un beigu laiku
14     System.out.println(System.currentTimeMillis()-ms);
15   }
16 }
```

Teorētiski:

ja 1GHz procesors 900 000 000 reizes darbinās ciklu, tad nepieciešamais laiks >=$\frac{900\,000\,000}{1\,000\,000\,000}$=**0.9s**

Eksperimentāli – nolasot sistēmas laiku milisekundēs

Datoram ar 1.8GHz procesoru – apmēram 920 milisekundes jeb **0.92s**, taču var būt vēl lēnāk, atkarībā no darbībām, ko veic ciklā.

IZMANTOTIE MATERIĀLI

Pieraksti no LU MII vadošā pētnieka Mārtiņa Opmaņa skaidrojuma skolotāju kursos par uzdevuma «Labi sakārtoti skaitļi» risinājumu. http://spoki.tvnet.lv/mistika/Kritosas-zvaigznes/186825.