

Ministerul Educației, Culturii și Cercetării



Departamentul Ingineria Software și Automatică

# RAPORT

*La structuri de date și algoritmi*

Lucrarea de laborator nr. 7

*Varianta 18*

A efectuat:

st. gr. TI-206

Cătălin Pleșu

A verificat:

Lector universitar

Vitalie Mititelu

**Chișinău 2021**

## **Tema: Tehnici de Programare**

### **Scopul:**

utilizarea diverselor tehnici de programare pentru scrierea programelor complexe.

### **Sarcina:**

18. Un om dorește să urce o scară cu  $N$  trepte. El poate urca la un moment dat una sau două trepte. Precizați în câte moduri poate urca omul scara.

### **Rezumat la temă:**

**Algoritmii forței brute** sunt exact cum sună - metode simple de rezolvare a unei probleme care se bazează pe puterea de calcul pură și pe încercarea tuturor posibilităților, mai degrabă decât a tehnicilor avansate pentru a îmbunătăți eficiența.

**Divide et Impera** este o metodă ce constă în:

- descompunerea problemei ce trebuie rezolvată într-un număr de subprobleme mai mici ale aceleiași probleme
- rezolvarea succesivă și independentă a fiecăreia dintre subprobleme
- recompunerea subsoluțiilor astfel obținute pentru a găsi soluția problemei inițiale.

**Programarea dinamică**, ca și metoda divide et impera, rezolvă problemele combinând soluțiile subproblemelor. După cum am văzut, algoritmii divide et impera partiționează problemele în subprobleme independente, rezolvă subproblemele în mod recursiv, iar apoi combina soluțiile lor pentru a rezolva problema inițială. Dacă subproblemele conțin subsubprobleme comune, în locul metodei divide et impera este mai avantajos de aplicat tehnica programării dinamice.

Cod sursă:

## Testarea programului:

Ca referință la testarea programului se va utiliza funcția ce calculează numărul de metode de urcare a scării prin forța brută.

```
[catalin 117] make  
gcc trepte.c  
./a.out  
SDA LAB7 - calcularea modurilor de a parcurge o scara de N trepte
```

Meniul:

```
1. forta bruta  
2. divide and conquer (limita 13)  
3. programare dinamica (fibonacci) (limita 47)  
4. alt numar de scari  
0. iesire
```

O scară de 1 treaptă poate fi urcată într-un singur mod, una de 2 în 2 moduri, o scara de 3 trepte în 3, scara de 4 în 5 moduri.

Pe prima linie este reprezentat numărul de scări:

1	:		1
---	---	--	---

1	:		1	1
2	:		2	

1	:		1	1	1
2	:		2	1	
3	:		1	2	

1	:		1	1	1	1
2	:		2	1	1	
3	:		1	2	1	
4	:		1	1	2	
5	:		2	2		

Am ajuns la concluzia că numărul de moduri de a urca scara corespunde elementului din numărului lui fibonacci cu indicele n, Cu exceptia elementului 0, fiind că așa scară nu poate exista.

Pentru a urca o scara de 2 trepte există 3 posibilități ceea ce este ilustrat in cazul forței brute și este confirmat și de celelalte metode.

### Setul 1 de date

Pentru o scară de 6 trepte există  $n$  modalități de a o urca:

	7	:		2	2	1	1
1	:		1	1	1	1	1
2	:		2	1	1	1	1
3	:		1	2	1	1	1
4	:		1	1	2	1	1
5	:		1	1	1	2	1
6	:		1	1	1	1	2
	8	:		2	1	2	1
	9	:		2	1	1	2
	10	:		1	2	2	1
	11	:		1	2	1	2
	12	:		1	1	2	2
	13	:		2	2	2	

Rezultatul obținut prin forța brută:

```
scara poate fi urcata in 13 moduri  
timpul de executie: 0.000078 sec.
```

Rezultatul obținut prin metoda divide and conquer:

```
optiunea - 2  
unu 6; doi 0; 1  
unu 4; doi 1; 5  
unu 2; doi 2; 6  
unu 0; doi 3; 1  
scara poate fi urcata in 13 moduri  
timpul de executie: 0.000028 sec.
```

Rezultatul obținut prin ultima metodă:

```
optiunea - 3  
scara poate fi urcata in 13 moduri  
timpul de executie: 0.000001 sec.
```

## Setul 2 de date

Pentru o scară de **10** trepte există **n** modalități de a o urca:

optiunea - 1										
1	:		1	1	1	1	1	1	1	1
2	:		2	1	1	1	1	1	1	1
3	:		1	2	1	1	1	1	1	1
4	:		1	1	2	1	1	1	1	1
5	:		1	1	1	2	1	1	1	1
6	:		1	1	1	1	2	1	1	1
7	:		1	1	1	1	1	2	1	1
8	:		1	1	1	1	1	1	2	1
9	:		1	1	1	1	1	1	1	2
10	:		1	1	1	1	1	1	1	2

11	:		2	2	1	1	1	1	1	1
12	:		2	1	2	1	1	1	1	1
13	:		2	1	1	2	1	1	1	1
14	:		2	1	1	1	2	1	1	1
15	:		2	1	1	1	1	2	1	1
16	:		2	1	1	1	1	1	2	1
17	:		2	1	1	1	1	1	1	2
18	:		1	2	2	1	1	1	1	1
19	:		1	2	1	2	1	1	1	1
20	:		1	2	1	1	2	1	1	1
21	:		1	2	1	1	1	2	1	1
22	:		1	2	1	1	1	1	2	1
23	:		1	2	1	1	1	1	1	2
24	:		1	1	2	2	1	1	1	1
25	:		1	1	2	1	2	1	1	1
26	:		1	1	2	1	1	2	1	1
27	:		1	1	2	1	1	1	2	1
28	:		1	1	2	1	1	1	1	2
29	:		1	1	1	2	2	1	1	1
30	:		1	1	1	2	1	2	1	1
31	:		1	1	1	2	1	1	2	1
32	:		1	1	1	2	1	1	1	2
33	:		1	1	1	1	2	2	1	1
34	:		1	1	1	1	2	1	2	1
35	:		1	1	1	1	2	1	1	2
36	:		1	1	1	1	1	2	2	1
37	:		1	1	1	1	1	2	1	2
38	:		1	1	1	1	1	1	2	2

39	:		2	2	2	1	1	1	1	
40	:		2	2	1	2	1	1	1	
41	:		2	2	1	1	2	1	1	
42	:		2	2	1	1	1	2	1	
43	:		2	2	1	1	1	1	2	
44	:		2	1	2	2	1	1	1	
45	:		2	1	2	1	2	1	1	
46	:		2	1	2	1	1	2	1	
47	:		2	1	2	1	1	1	2	
48	:		2	1	1	2	2	1	1	
49	:		2	1	1	2	1	2	1	
50	:		2	1	1	2	1	1	2	
51	:		2	1	1	1	2	2	1	
52	:		2	1	1	1	2	1	2	
53	:		2	1	1	1	1	2	2	
54	:		1	2	2	2	1	1	1	
55	:		1	2	2	1	2	1	1	
56	:		1	2	2	1	1	2	1	
57	:		1	2	2	1	1	1	2	
58	:		1	2	1	2	2	1	1	
59	:		1	2	1	2	1	2	1	
60	:		1	2	1	2	1	1	2	
61	:		1	2	1	1	2	2	1	
62	:		1	2	1	1	2	1	2	
63	:		1	2	1	1	1	2	2	
64	:		1	1	2	2	2	1	1	
65	:		1	1	2	2	1	2	1	
66	:		1	1	2	2	1	1	2	
67	:		1	1	2	1	2	2	1	
68	:		1	1	2	1	2	1	2	
69	:		1	1	2	1	1	2	2	
70	:		1	1	1	2	2	2	1	
71	:		1	1	1	2	2	1	2	
72	:		1	1	1	2	1	2	2	

74	:		2	2	2	2	1	1		
75	:		2	2	2	1	2	1		
76	:		2	2	2	1	1	2		
77	:		2	2	1	2	2	1		
78	:		2	2	1	2	1	2		
79	:		2	2	1	1	2	2		
80	:		2	1	2	2	2	1		
81	:		2	1	2	2	1	2		
82	:		2	1	2	1	2	2		
83	:		2	1	1	2	2	2		
84	:		1	2	2	2	2	1		
85	:		1	2	2	2	1	2		
86	:		1	2	2	1	2	2		
87	:		1	2	1	2	2	2		
88	:		1	1	2	2	2	2		
89	:		2	2	2	2	2			

1. scara poate fi urcata in 89 moduri  
timpul de executie: 0.000410 sec.
2. scara poate fi urcata in 89 moduri  
timpul de executie: 0.000033 sec.
3. scara poate fi urcata in 89 moduri  
timpul de executie: 0.000001 sec.

### Setul 3 de date

Pentru o scară de 30 trepte există  $n$  modalități de a o urca:

1. scara poate fi urcata in 1346269 moduri  
timpul de executie: 8.832981 sec.
3. scara poate fi urcata in 1346269 moduri  
timpul de executie: 0.000002 sec.

### Concluzii:

1. Am rezolvat aceeași problemă prin mai multe tehnici de programare, observând că unele tehnici sunt mai optime decât altele.
2. Algoritmii ce utilizează forța brută sunt cei mai ineficienți algoritmi, deși în cazul dat este singura opțiune de a vizualiza modalitățile de a urca scara respectivă.
3. Am observat că numărul de moduri de a urca scara corespunde elementului din numărului lui fibonacci cu indicele  $n$ , de aici rezultă ce cea mai eficientă metodă de rezolvare a problemei are o complexitate liniară.
4. Numărul de posibilități crește atât de rapid încât în tipul de date integer încap doar posibilitățile până la 47 de trepte.