

Backtracking, Generarea  
permutarilor unei multimi

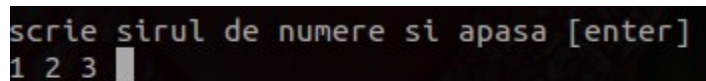
## 1 Folosirea programului

Programul generează toate permutările unei mulțimi de caractere introduse de utilizator având la bază paradigma backtracking. Mulțimea de numere va fi introdusă la tastatură și poate conține un maximum de 80 de caractere.

Programul nu-l constrânge pe utilizator să folosească doar numere. Bazându-se pe un algoritm generalist, programul permite generarea permutărilor unei mulțimi în care elementele nu sunt omogene.

## 2 Folosirea programului

Odată lansat, programul îi va cere utilizatorului să introducă șirul de caractere a cărui permutări dorește să le afișeze programul (fig. 1). Utilizatorul va trebui să introducă elementele mulțimii. Fiecare element va fi separat de unul sau mai multe spații albe sau de unul sau mai multe taburi.

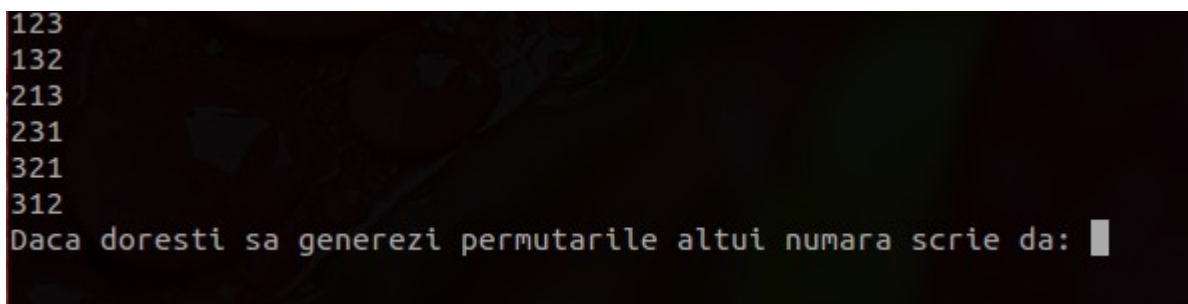


```
scrie sirul de numere si apasa [enter]
1 2 3 █
```

Fig. 1 – Introducerea datelor

Programul poate citi un maximum de 80 de caractere pentru fiecare mulțime, în acest număr includem și caracterele folosite drept separatoare. Prin urmare, cu cât se introduc mai multe caractere separatoare, cu atât mai puține elemente va putea conține mulțimea. Programul nu interpretează virgula ca fiind un caracter separator valid, folosirea ei poate duce la comportament imprevizibil.

După ce a fost introdusă mulțimea programul va afișa pe fiecare linie câte o permutare a acelei mulțimi și îl va întreba pe utilizator dacă dorește să ge



```
123
132
213
231
321
312
Daca doresti sa generezi permutarile altui numara scrie da: █
```

Fig. 2 – Generare a outputului

Pentru a genera permutarile altei multimi utilizatorul va scrie “da”, in caz contrar programul isi va termina executia.

### 3 Detalii de implementare

Programul a fost scris in limbajul de programare C si foloseste standardul **ISO/IEC 9899:1999**. Toate functionalitatile programului au fost implementate folosind biblioteca standard, prin urmare programul este unul protabil. Acesta a fost testat pe platformele Windows 8 si Ubuntu Linux 14.04.

#### 3.1 Compilarea programului

Pentru compilarea programului am folosit compilatorul **GNU GCC**. Sub sistemele pe baza de Linux acesta vine preinstalat, pe Windows se va instala MinGW(Minimalist GNU for Windows) ce este un port al GCC pentru Windows.

Pentru a compila programul va trebui sa executam comanda:

```
gcc -Wall -std=c99 main.c
```

#### 3.2 Descrierea codului sursa

Programul este unul modular, fiind impartit in subprograme ce implimenteaza functionalitatile principale.

Odata pornit, programul va apela functia author() ce va afisa detalii despre autor, profesorul coordonator si scoala de provenienta. Dupa apelul la aceasta functie, fluxul de executie va intra intr-o bucla do-while ce va rula pana cand valoarea vectorului message nu va mai fi egala cu da, caz in care programul isi opreste executia. Am ales sa folosesc o bucla do-while pentru a simplifica modul de functionarea a programului in care se doreste executarea unei actiuni de mai multe ori si pentru a ma asigura ca programul calculeaza permutarile a cel putin unei multimi.

Dupa ce am apelat functia printf ce ii va afisa utilizatorului instructiuni despre introducerea outputului, se va apela functia readString ce va citi un string de maximum 80 de caractere si-l va salva in bufferul a.

Apoi se apeleaza functia permute pasandu-i ca si argumente bufferul a indexul de inceput al acestuia, respectiv 0, si indexul ultimului caracter din string, respectiv strlen(a)-1.

Intr-un final, folosindu-ne de functia scanf citim raspunsul urilizatorului la inrebarea afisa de apelul la functia printf, realizat cu o linie de cod mai sus.

Daca valoarea bufferului message este egala cu “da” bucla va incepe o noua iteratie, in caz contrar programul isi va termina executia.