**Laborator 12**

**Backtracking recursiv**

## **Obiectiv**

* Aprofundarea **metodei de căutare cu revenire**

## **Aspecte teoretice**

**Problemă.**

Între insulele unui arhipelag se află **n** poduri. Un turist îşi propune să viziteze aceste poduri, o singură dată, pornind de la o anumită insulă. Să se afişeze toate posibilităţile de parcurgere a celor n poduri.

**Rezolvare**

Se porneşte analiza problemei prin identificarea celor mai potrivite structuri de date care să modeleze conţinutul problemei.

**Date de intrare**

n - numărul de poduri

ins\_start – insula de la care se porneşte

pod[n] – un vector de structuri de dimensiune n, care descrie fiecare pod şi are ca membri două numere întregi reprezentând cele două insule (ins1, ins2), pe care le leagă.

## Date de ieşire

Şirul x1, x2, ...xn, fiecare element xi reprezentând indicele unui pod [0,n-1].

## Procedura propusă pentru rezolvarea problemei

Pentru a mări viteza de execuţie a procedurii recursive, se va considera că variabila **n** şi tablourile **x[]** şi **pod[]** sunt globale.

La fiecare pas se testează dacă podul **i** , cu **i=0,n-1** , se poate concatena soluţiei parţiale determinate până în acel moment.

**procedura** Plimbare(insula\_crt, k) **este**

 **daca** n==k **atunci**

\*) scrie x[0],...,x[n-1]

 **altfel**

 **pentru** i=0,n-1 **execută**

**daca** POSIBIL(i,k,insula\_crt) **atunci**

x[k]=i

**daca** insula\_crt==pod[i].ins1 **atunci**

ins=pod[i].ins2

 **altfel**

 ins=pod[i].ins1

 @

 Plimbare (ins, k+1)

 @

@

 @

**sfarsit**

După ce se include la pasul **k** un pod în soluţia parţială, se determină noua **ins\_crt** pentru pasul următor (acel capăt al podului ales la pasul **k** ce nu se află pe vechea **ins\_crt**).

Predicatele de extindere a soluţiei la pasul **k**:

 **POSIBIL (alfa, k, ins\_crt)**

Vor verifica dacă:

* podul alfa nu a fost vizitat (alfa != x[i] pentru i=0,k-1)
* insula curentă este una din extremităţile podului alfa (pod[alfa].ins1 == ins\_crt sau pod[alfa].ins2 == ins\_crt )

**functie** POSIBIL (alfa, k, ins\_crt) **este**

 **pentru** j=0,k-1 **executa**

 **daca** x[j] == alfa **atunci**

 **intoarce** FALS

 @

 @

**intoarce** pod[alfa].ins1 == ins\_crt **sau** pod[alfa].ins2 == ins\_crt

În programul principal rezolvarea problemei va fi declanşată prin apelul:

**execută** Plimbare (ins\_start, 0)

## **Tema de realizat în timpul laboratorului**

## Exemplu subiect examen

Scrieti un program care sa rezolve problema vizitarii podurilor o singura data utilizand algoritmul backtracking sau un alt algoritm la alegerea dumneavoastră. Veţi lua în considerare faptul că datele de intrare vor fi preluate dintr-un fişier text cu n+2 înregistrări, având următorul format:

n

ins\_start

ins\_11 ins\_12

ins\_21 ins\_22

...

ins\_n1 ins\_n2

**Observaţie.** Toate valorile din fişier sunt întregi.

Programul implementat va afisa urmatorul meniu:

|  |  |
| --- | --- |
| Optiune meniu | Punctaj |
| C - citeste date de la tastatura | 1p |
| F - citeste date din fisier | 1p |
| A - afiseaza matricea drumurilor | 1p |
| R - rezolva problema | 5p (cate 1p pentru fiecare set de date de test corect rezolvat:* Un set de date citit de la Tastatura
* 4 seturi de date citite din fisier)
 |
| I - info autor | 0.5p |
| T - termina | 0.5p |
| Oficiu | 1p  |

Observatii:

1. Fisierele de test contin si date eronate sau date care nu conduc la o solutie. Implementati programul astfel incat să afişeze mesajele **Date eronate** sau **Fără soluţie**, dacă nu sunt date valide sau nu este posibilă găsirea unei soluţii în condiţiile cerute.
2. **BONUS**. Rescrieţi programul astfel încât să determine insula de pe care turistul să plece şi să revină pe aceeaşi insulă după ce a parcurs toate podurile o singură dată.

## **Teme pentru studiu individual**

Veţi rezolva cel puţin 2 probleme indicate de cadrul didactic.

1. La o Masă rotundă privind “Creşterea rolului întreprinderilor în finanţarea învăţământului superior” au fost invitate să trimită câte un reprezentant **n** firme. Este cunoscut faptul că fiecare firmă invitată are una sau mai multe firme concurente printre celelalte invitate. Scrieţi un program care să aşeze participanţii la o masa rotundă astfel încât reprezentanţii a două firme concurente să nu stea alături.
2. Între **n** oraşe exista o reţea de şosele care unesc două oraşe între ele. Să se determine toate posibilităţile de alegere a şoselelor, astfel încât sa se ajungă din oraşul **s** în oraşul **d**.
3. Un student la filologie poseda **n** dicţionare bilingve care permit traducerea dintr-o limba **i** intr-o limba **j** cu **1<=i, j<=n**. Sa se determine toate seturile de dicţionare care îi permit studentului să realizeze o traducere între limba **s** şi limba **d**.
4. Pe un deal, într-un punct oarecare, se află o minge. Să se determine toate drumurile pe care poate coborî mingea astfel încât să ajungă la poalele dealului.
5. Sa se acopere un hol de lungime **L** cu un număr minim de bucăţi de mochete de lungime **li** existente în depozit.
6. Se consideră un labirint descris cu ajutorul unui tablou bidimensional. Să se redacteze programul care determină toate drumurile de ieşire din labirint.
7. Într-un grup de persoane, fiecare persoana se cunoaşte pe sine si cunoaşte eventual alte persoane din grup. Sa se formeze si sa se afişeze toate echipele posibile de persoane astfel încât, pentru o echipă, fiecare din cele **n** persoane sa fie cunoscută de cel puţin un membru al acelei echipe.
8. Sa se realizeze parcurgerea tabelei de şah de către un cal care pleacă dintr-o poziţie oarecare, astfel încât sa parcurgă fiecare câmp al tablei o singura dată, mai puţin câmpul de coordonate **(i,j).**
9. Un dresor trebuie sa scoată **m** lei si **n** tigri din arena, astfel încât sa nu scoată doi tigri unul după altul. Sa se genereze toate posibilităţile de înşiruire a leilor si tigrilor.
10. O caravană formată din **n** cămile călătoreşte prin deşert, în şir indian. Pentru a sparge monotonia zilelor lungi de drum, beduinul sef se hotărăşte sa schimbe aşezarea cămilelor, astfel încât fiecare cămilă sa nu mai vadă in fata ei aceeaşi cămila de până atunci. Sa se genereze toate posibilităţile de aşezare a cămilelor, cunoscând modul de aşezare din prima zi.
11. Un soldat trebuie sa parcurgă un teren minat pentru a ajunge in propriile linii. Sa se genereze toate drumurile posibile prin care soldatul ajunge nevătămat la camarazii săi.
12. Fie **n** persoane, **n** locuri de munca si **C(i,j)** salariul pretins de persoana **i** dacă este angajată la locul de munca **j**. Sa se determine modul optim de angajare a tuturor persoanelor.
13. Dintr-un grup de **n** persoane, dintre care **p** femei, trebuie formată o delegaţie de **k** persoane din care **l** femei. Să se precizeze toate delegaţiile care se vor forma.
14. Se consideră un labirint de formă dreptunghiulară având pereţi verticali şi orizontali. Să se găsească drumul cel mai scurt dintre două puncte date.
15. Fiind dat un carosaj dreptunghiular cu **m** linii şi **n** coloane, în care anumite poziţii sunt ocupate (interzise), se cere să se determine toate poziţiile în care poate ajunge un mobil ce pleacă din punctul iniţial **(i0, j0)**, ştiind că el se poate deplasa doar astfel:
	* cu o poziţie la dreapta pe aceiaşi linie;
	* cu o poziţie la stânga pe aceiaşi coloană;
	* cu o poziţie în sus pe aceiaşi coloană.
16. Care este numărul minim de regi care pot fi aşezaţi pe o tablă de şah, astfel încât să ţină sub ameninţare toate pătratele neocupate de figuri ?
17. Care este numărul maxim de cai care pot fi aşezaţi pe tabla de şah cu **n2** pătrate, astfel încât să nu se ameninţe?
18. Problema bilei. Un teren este reprezentat sub forma unei matrice cu **n** linii şi **n** coloane. Elementele **(i, j)** ce aparţin mulţimii numerelor reale reprezintă cotele diferitelor porţiuni din acest teren. Se presupune că o bilă se găseşte pe o porţiune având o anumită cotă. Se cere să se precizeze toate traseele posibile de a fi urmate de bilă spre a ieşi din teren învecinată cu o cotă strict inferioară cotei terenului pe care se găseşte bila.
19. Se consideră o mulţime formată din **n** persoane, în care fiecare se cunoaşte pe sine şi eventual, alte persoane. Să se afişeze grupurile care se pot forma cu aceste persoane, fiecare persoană din grup trebuind să cunoască toate celelalte persoane ale grupului. Relaţia “persoana i cunoaşte persoana j” este dată printr-o matrice 

*Observaţii:*

* + relaţia nu este în mod normal simetrică şi nici tranzitivă, deci Cij=Cji în anumite cazuri, iar din Cij =1 şi Cjk =1 nu rezultă Cik =1
	+ o persoană nu poate aparţine decât unui singur grup.
1. Se consideră **n** fete care urmează să se căsătorească într-un mod optim cu **n** băieţi. Fetele şi băieţii îşi exprimă preferinţele unul faţă de altul prin numere reale din intervalul [0, 1]. Preferinţa fetei **i** pentru băiatul **j** este dată de **fb[i, j]**, iar preferinţa băiatului **i** pentru fata **j** este dată de **bf[i, j]**; elementele matricilor **fb** şi **bf** sunt numere reale din intervalul [0, 1]. Băiatul ales de fata **i** are numărul **x[i]**. Bineînţeles, vectorul **x** va fi un vector de permutare. Costul căsătoriei fetei **i** cu băiatul **x[i]** este **fb [i, x[i]] \* bf[x [i], i]**, iar costul general (care trebuie maximizat) este suma după **i** a acestor valori. Mai mult, se cere ca cele **n** căsătorii cu i≠j astfel încât fata **i** să prefere băiatul **x[j]** băiatului **x[i]**, iar băiatul **x[j]** să prefere fata **i** fetei **j**.

(G. M. 7-8 /1993)