METODA GREEDY & PROBLEME DE LOGICA

METODA GREEDY

- Metoda Greedy este o metoda folosita in algoritmica care presupune ca la fiecare posibilitate de a alege o continuare, sa se aleaga metoda cea mai favorabil la acel moment.
- Greedy = (in traducere din engleza) lacom.
- Metoda Greedy este o metoda foarte generala care nu duce mereu la un rezultat final corect, dar exista si multe probleme care se rezolva corect prin aceasta metoda. Decizia de a putea folosii acest concept sta in mainile programatorului.

METODA GREEDY

- Metoda Greedy este un concept si nu un algoritm, acest lucru este foarte important de inteles pentru a nu incerca folosirea unui sablon la acest capitol.
- Daca la Backtracking si la alte capitole existau niste algoritmi care se putea modifica sa mearga pe toate problemele, aici lucrurile stau usor diferit, intrucat modul de abordare al problemelor ar trebui sa fie unul natural.
- Cea mai usoara metoda de a concepe un algoritm de acest gen este simularea metodei naturale de a rezolva problema aceasta pe hartie.

METODA GREEDY

- Una dintre cele mai importante faze a rezolvarii unei probleme prin metoda Greedy este proiectarea algoritmului. Asa cum spuneam, aceasta metoda este una dintre cele mai naturale metode, intrucat presupune punerea in cod a rationamentului uman

- In general, se citeste problema si se incearca o rezolvare in cap a acesteia. Daca se ajunge la rezultatul final corect prin rationamentul uman, abordarea a fost una buna si vom incerca punerea in cod a acestuia.

UN EXEMPLU RELEVANT

- La aceasta problema, e natural ca pentru a maximiza numarul de masini care se repara, sa sortam masinile crescator dupa timpii de reparere si astfel sa vedem cate masini putem repara in timpul dat.

Cerinţa

Cunoscând timpul necesar pentru repararea fiecărei maşini, scrieţi un program care calculează numărul maxim de maşini care pot fi reparate într-un interval de timp `T`.

Date de intrare

Pe prima linie a fişierului `masini.in` se găsec două numere naturale `n` şi `T` separate printr-un singur spaţiu, reprezentând numărul de maşini din curtea atelierului auto şi timpul total în care se va lucra.

Pe linia a doua, separate prin câte un spaţiu, se găsesc `n` numere naturale t_1, t_2, ..., t_n, reprezentând timpii necesari pentru repararea fiecărei maşini.

Date de ieşire

Pe prima linie a fişierului `masini.out` se va găsi un număr natural `k`, reprezentând numărul maxim de maşini care pot fi reparate.

Restricţii şi precizări

- `1 < n, T <= 1000`
- numerele de pe a doua linie a fişierului de intrare vor fi mai mici sau egale cu `100`

Exemplu

`masini.in`

```
5 10
6 2 4 8 2
```

`masini.out`

```
3
```

Explicaţie

SOLUTIA:

```cpp
#include <fstream>
using namespace std;

ifstream cin("masini.in");
ofstream cout("masini.out");

// Greedy

int n, t;
int a[1001];

void sortare(int a[], int n){
    bool este_sortat = false;
    while(!este_sortat){
        este_sortat = true;
        for(int i = 1; i < n; ++i)
            if(a[i] > a[i+1]){
                este_sortat = false;
                swap(a[i], a[i+1]);
            }
    }
}
```

```cpp
int main(){
    cin >> n >> t;
    for(int i = 1; i <= n; ++i)
        cin >> a[i];
    sortare(a, n);
    int i = 1;
    while(t - a[i] >= 0 && i <= n)
        t -= a[i], i++;
    cout << i - 1;
    return 0;
}
```

PROBLEMA "SPECTACOLE"

Cerința

La un festival sunt programate `n` spectacole. Pentru fiecare se cunoaște momentul de început și momentul de sfârșit, exprimate prin numere naturale. Un spectator dorește să urmărească cât mai multe spectacole în întregime.

Determinați numărul maxim de spectacole care pot fi urmărite, fără ca acestea să se suprapună.

Date de intrare

Fişierul de intrare `spectacole.in` conţine pe prima linie numărul `n`. Pe fiecare dintre următoarele `n` linii se află câte două numere naturale `X Y`, reprezentând momentul de început și momentul de sfârșit al unui spectacol.

Date de ieşire

Fişierul de ieşire `spectacole.out` va conţine pe prima linie numărul `S`, reprezentând numărul maxim de spectacole care pot fi urmărite, fără să se suprapună.

Restricţii şi precizări

- `1 ≤ n ≤ 100`
- momentele de început și sfârșit ale spectacolelor sunt numere naturale mai mici decât `1.000.000`
- pentru fiecare spectacol, `X < Y`
- dacă momentul de început al unui spectacol și momentul de sfârșit al altui spectacol coincid, pot fi urmărite ambele spectacole

SOLUTIE

- Vom sorta intervalele dupa ora de final. De ce? Pentru ca cu cat ies mai repede de la spectacolul i (ca timp), am oportunitatea de a alege din cat mai multe spectacole ulterioare.
- Exact cum am proceda si in viata reala, de asta se numeste GREEDY.

```cpp
#include <fstream>
using namespace std;

ifstream cin("spectacole.in");
ofstream cout("spectacole.out");

// Metoda Greedy

struct spectacol{
    int start, finish;
}a[101];

int n;

void sortare_spectacole(spectacol a[], int n){
    for(int i = 1; i < n; ++i)
        for(int j = i + 1; j <= n; ++j)
            if(a[i].finish > a[j].finish)
                swap(a[i], a[j]);
}

int main(){
    cin >> n;
    for(int i = 1; i <= n; ++i)
        cin >> a[i].start >> a[i].finish;
    sortare_spectacole(a, n);
    int cnt = 1;
    int ora_fin = a[1].finish;
    for(int i = 2; i <= n; ++i)
        if(ora_fin <= a[i].start){
            cnt++;
            ora_fin = a[i].finish;
        }
    cout << cnt;
    return 0;
}
```

SEAMANA CU CEVA CUNOSCUT ACEASTA GRILA DIN 2021?

15. Se dă o mulțime ***S***, care conține ***n*** intervale specificate prin capătul stâng s_i și capătul drept d_i ($s_i < d_i \; \forall \; i = 1...n$). Se dorește determinarea unei submulțimi $S' \subseteq S$ de ***m*** elemente, astfel încât să nu existe două intervale în ***S'*** care se intersectează și ***m*** să aibă cea mai mare valoare posibilă.
Care dintre următoarele strategii rezolvă corect problema?

A. Se sortează intervalele din mulțimea ***S*** crescător după capătul stâng. Se adaugă primul interval din șirul sortat în ***S'***. Se parcurg celelalte elemente din șir în ordinea sortată și când se întâlnește un interval care nu se intersectează cu intervalul care a fost adăugat ultima oară în ***S'***, se adaugă și acesta în ***S'***.
B. Se sortează intervalele din mulțimea ***S*** crescător după capătul drept. Se adaugă primul interval din șirul sortat în ***S'***. Se parcurg celelalte elemente din șir în ordinea sortată și când se întâlnește un interval care nu se intersectează cu intervalul care a fost adăugat ultima oară în ***S'***, se adaugă si acesta în ***S'***.
C. Se sortează intervalele din mulțimea ***S*** crescător după lungimea intervalului. Se adaugă primul interval din șirul sortat în ***S'***. Se parcurg celelalte elemente din șir în ordinea sortată și când se întâlnește un interval care nu se intersectează cu intervalul care a fost adăugat ultima oară în ***S'***, se adaugă și acesta în ***S'***.
D. Se sortează intervalele din mulțimea ***S*** crescător după numărul de intervale din ***S*** cu care se intersectează. Se adaugă primul interval din șirul sortat în ***S'***. Se parcurg celelalte elemente din șir în ordinea sortată și când se întâlnește un interval care nu se intersectează cu intervalul care a fost adăugat ultima oară în ***S'***, se adaugă și acesta în ***S'***.

PROBLEMELE DE LOGICA

- In cadrul grilelor de la Examenul de Admitere la UBB, multe probleme presupun utilizarea unui rationament logic foarte bun.
- Inclusiv problemele de la metoda Greedy pot fi considerate probleme de logica.
- Orice problema care nu poate fi interpretata prin prisma unui algoritm, trebuie vazuta ca o problema de logica, iar cel mai bun raspuns la o astfel de intrebare este dat daca reusim sa rezolvam problema utilizand rationamentul nostru.

HAIDETI SA INCERCAM SA REZOLVAM GRILA URMATOARE:

23. Fie ***s*** un șir de numere naturale unde elementele s_i sunt de forma $s_i = \begin{cases} x, & \text{dacă } i = 1 \\ x + 1, & \text{dacă } i = 2 \\ s_{(i-1)} @ s_{(i-2)} & \text{dacă } i > 2 \end{cases}$,

(***i*** = 1, 2, ...). Operatorul @ concatenează cifrele operandului stâng cu cifrele operandului drept, în această ordine (cifre aferente reprezentării în baza 10), iar ***x*** este un număr natural ($1 \leq x \leq 99$). De exemplu, dacă ***x*** = 3, șirul ***s*** va conține valorile 3, 4, 43, 434, 43443, ... Precizați numărul cifrelor acelui termen din șirul ***s*** care precede termenul format din ***k*** ($1 \leq k \leq 30$) cifre.

A. dacă ***x*** = 15 și ***k*** = 6, numărul cifrelor termenului aflat în șirul ***s*** în fața termenului format din ***k*** cifre este 5.
B. dacă ***x*** = 2 și ***k*** = 8, numărul cifrelor termenului aflat în șirul ***s*** în fața termenului format din ***k*** cifre este 5.
C. dacă ***x*** = 14 și ***k*** = 26, numărul cifrelor termenului aflat în șirul ***s*** în fața termenului format din ***k*** cifre este 16.
D. dacă ***x*** = 5 și ***k*** = 13, numărul cifrelor termenului aflat în șirul ***s*** în fața termenului format din ***k*** cifre este 10.

CE SPUNETI DE ACEASTA?

30. O matrice cu 8 linii, formată doar din elemente 0 și 1, are următoarele trei proprietăți:

a. prima linie conține un singur element cu valoarea 1,
b. linia j conține de două ori mai multe elemente nenule decât linia $j - 1$, pentru orice $j \in \{2, 3, \ldots, 8\}$,
c. ultima linie conține un singur element cu valoarea 0.

Care este numărul total de elemente cu valoarea 0 din matrice?

A. 777
B. 769
C. 528
D. nu există o astfel de matrice

O PROBLEMA MAI INTERESANTA…

2. Un fișier Excel conține ***n*** înregistrări numerotate de la 1 la ***n***. Aceste înregistrări trebuie copiate într-un fișier Word în care înregistrările se vor aranja în câte ***r*** rânduri și ***c*** coloane pe fiecare pagină (cu excepția primei și ultimei pagini). Pe prima pagină a documentului Word, datorită prezenței unui antet, numărul de rânduri este r_1, $r_1 < r$ (numărul de rânduri prezent pe prima pagina este mai mic).

Înregistrările vor fi aranjate în fișierul Word pe fiecare pagină de sus în jos pe fiecare coloană, coloanele fiind completate de la stânga la dreapta: dacă prima înregistrare de pe o pagină are numărul de ordine ***i***, înregistrarea cu numărul de ordine ($i + 1$) va fi prezentă sub ea, iar înregistrarea cu numărul de ordine (**i** + **r**) va fi prima înregistrare de pe coloana 2 de pe pagina respectivă ș.a.m.d.

Pentru $n = 5000$, $r = 46$, $r_1 = 12$ și $c = 2$ pe ce pagină a documentului Word și pe ce coloană se va regăsi înregistrarea cu număr de ordine $i = 3245$?

A. Pagina 36, ultima coloană
B. Pagina 37, prima coloană
C. Pagina 37, ultima coloană
D. Pagina 38, prima coloană

ORGANIZAREA GANDIRII

- Probabil cel mai important lucru cand rezolvam o problema de logica este sa ne organizam bine informatia in creier.
- De asemnea, daca vrem ca lucrurile sa ne iasa foarte bine, este extrem de important sa nu ne grabim si sa gasim cele mai bune metoda in care noi ne putem maximiza potentialul de a gasi raspunsul.
- Aceste grile sunt printer preferatele mele deoarece nu tin de cunostiintele de informatica cat tin de capacitatea creierului de a face conexiuni simple.
- Consider ca pentru a putea rezolva genul acesta de problema, gandirea critica este un factor esential, lucru necesar si pentru o cariera de succes ca programator.

CIORNA 1 (SLIDE 12)

b) • l_1: 1 element 1

l_2: 2 elem. 1

l_3: 4 elem. 1

⋮

l_8: 2^{8-1} elem 1 = 128 elem. 1 (1)

$S = 1+2+\dots+128 = 2^8 - 1 = 255$

$129 \cdot 8 = 1032$

c) l_8 - are un elem 0 (2)

Din (1) și (2) $\Rightarrow$ 128+1 coloane = 129

Avem $\overbrace{1+2+4+\dots+128}^{S}$ elem 1 și

$129 \cdot 8$ elemente în total $\Rightarrow$

Avem $129 \cdot 8 - S$ elemente 0

$\Rightarrow$ Avem 777 elemente 0

CIORNA II (SLIDE 13)

$i = 3245$ $\quad \lambda = 46$ $\quad \lambda_1 = 12$ $\quad C = 2$

$p_1 : 12 \cdot 2 = 24$

$p_2 \ldots \infty : 46 \cdot 2 = 92$

Umplem prima pagină și rămân $3245 - 24 = 3221$ înregistrări

$3221 \mid 92$

$276 \quad\quad 35$, $r = 1$

-461

-460

$= 1$

Numărul de pagini pline este $35 + 1$ și mai rămâne o înregistrare

VA MULTUMESC!

Somesan Paul-Ioan