Tekwill.
În Fiecare Școală

MONEDE LACOME

CURS: PROGRAMAREA ALGORITMILOR C/C++

MENTOR: GAIDĂU MIHAI

CURSANT: MIHĂLACHE LILIA

Problemă

La un magazin specializat managerul a fixat restricții casierului să dea rest clienților, care dau o sumă oarecare de bani pentru cumpărături, un număr cât mai mic de monezi.

Presupunem că casierul are următoarea listă disponibilă de monede: 50lei, 20lei, 10lei, 5lei, 1leu.

Moneda unitate este necesar să se regăsească printre ele.

Soluție:

- se sortează descrescător vectorul de monede;
- atâta timp cât suma este diferită de zero se determină ce monede se folosesc și câte astfel de monede.

Pentru rezolvare:

Tehnica Gredy

Tehnica Greedy

In funcție de specificul problemei, un algoritm greedy poate conduce la soluția optimă sau la o soluție destul de bună, deși suboptimală. Rezultatul unui algoritm greedy pentru o problemă data depinde și de datele concrete ale problemei, sau chiar de ordinea introducerii lor.

Prezentarea generală

De exemplu, în problema noastră monede lacome, sau sume de bani printr-un numar minim de monede de valori date, rezultatul (optim sau suboptim) depinde de valorile monedelor și de valoarea sumei. Algoritmul **Greedy** foloseste monedele în ordinea descrescatoare a valorilor lor, deci **“se repede**” la monedele de valoare maximă, care vor fi în număr mai mic pentru aceeași sumă.
Fie monede de valori **11, 5 si 1**:
– pentru suma **12** rezultatul algoritmului greedy va fi optim (două monede de valori: **11 si 1**)
– dar pentru suma **15** rezultatul algoritmului greedy nu va fi optim (5 monede de valori: **11, 1, 1, 1, 1,** în loc de 3 monede de valori **5, 5, 5**).

Schema generala a metodei

Pentru a exemplifica această metodă considerăm o mulţime A cu n elemente. Problema care ar trebui rezolvată, constă din determinarea unei submulţimi B a lui A. Aceasta trebuie să îndeplinească anumite condiţii pentru a fi acceptată ca soluţie.

Dintre toate submulţimile acceptate (numite soluţii posibile), se va alege una singură numită soluţie optimă.

Schema metodei:

```
Ord_Desc(n,b)
i ← 0
cât timp (S > 0) şi (i < n) execută
   i ← i+1
   x_i ← [S/b_i]
   nrb ← nrb+x_i
   S ← S -x_i*nrb
sfârşit cât timp
dacă S = 0 atunci
   scrie x_1, ..., x_i
altfel
   scrie 'Nu s-a gasit solutie.'
sfârşit dacă
sfârşit subalgoritm
```

Metoda Bubble sort

Metoda bulelor se bazează pe următoarea idee:

- fie un vector Monede[i] cu n elemente
- parcurgem vectorul și pentru oricare două elemente învecinate care nu sunt în ordinea dorită, le interschimbăm valorile
- după o singură parcurgere, vectorul nu se va sorta, dar putem repeta parcurgerea
- dacă la o parcurgere nu se face nicio interschimbare, vectorul este sortat

Ilustrare

Bubble Sort

TUTORIAL PROBLEMS VISUALIZER BETA

INPUTS

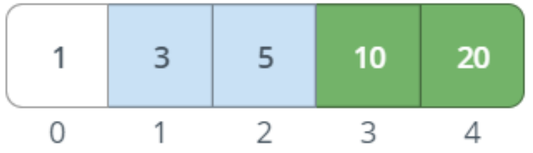
Array size: 5

Array layout: Array

Array Values (optional): 20 5 1 10 3

Run Reset

VISUALIZE



Secvența de cod

```
do
{
  sortat = true;
  for(int i = 0 ; i < n - 1 ; i ++)
    if(monede[i] < monede[i+1])
    {
      int aux = monede[i];
      monede[i] = monede[i+1];
      monede[i+1] = aux;
      sortat = false;
    }
}
while(!sortat);
```

Rezolvare

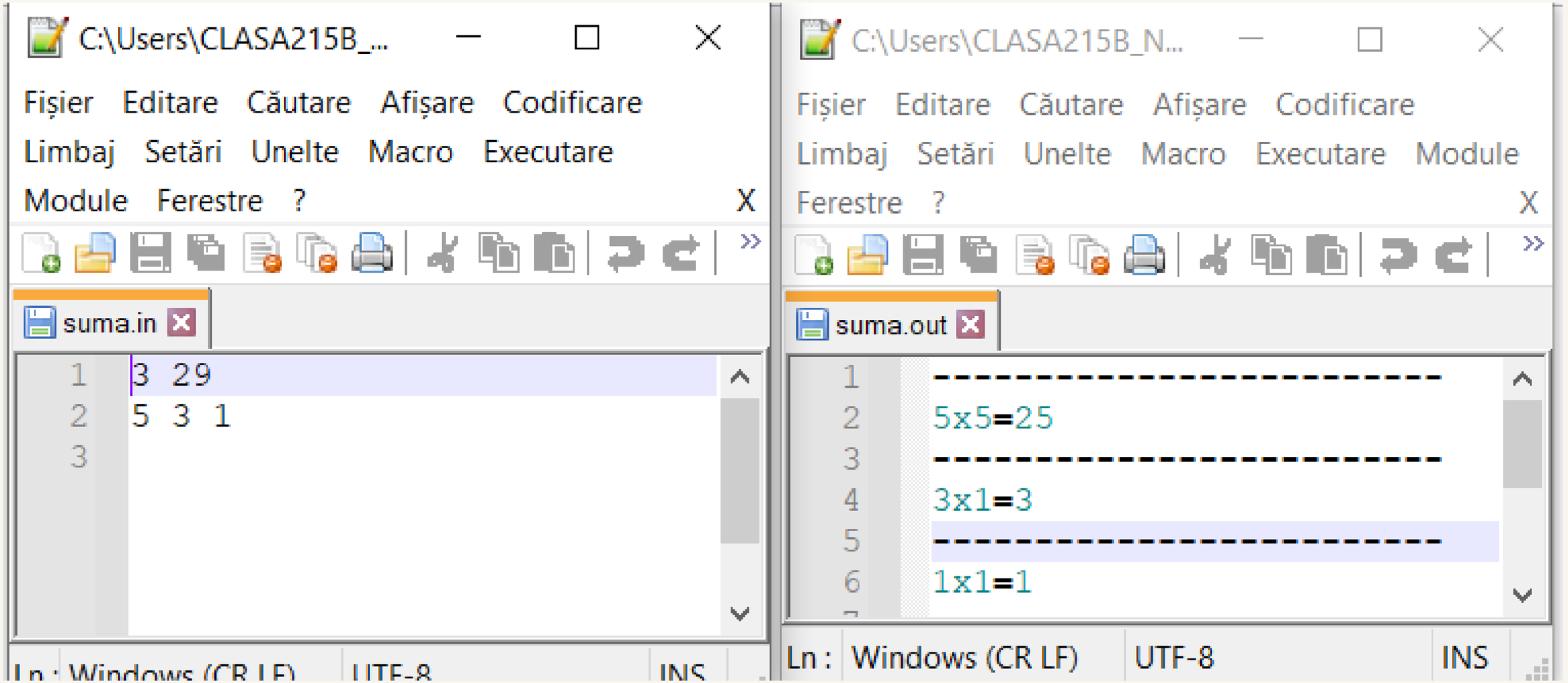
```cpp
#include <fstream>
using namespace std;
ifstream in("suma.in");
ofstream out("suma.out");

int main()
{
    int monede[100], nr_monede, i,sortat, schimb, suma;
//citim in fisier numarul de monede si suma
    in>>nr_monede>>suma;
// citim valoarea monedelor cu care se va restitui suma ce trebuie returnata
    for(i=1; i<=nr_monede; i++)
        in>>monede[i];
//utilizand o metoda se sortare se sorteaza monedele in ordine descrescatoare
    do
    {
        sortat = 0;
        for(i = 1; i<nr_monede; i++ )
            if(monede[i]<monede[i+1])
            {
            schimb = monede[i];
            monede[i] = monede[i+1];
            monede[i+1] = schimb;
        sortat = 1;
    }
    }

    while (sortat == 1);
    i = 1;

//utilizam metoda Greedy pentru calcularea numarului de monezi restituite
    while(suma!=0)
    {
        if (suma/monede[i])
        {
            out<<"-------------------------"<<endl;
            out<<monede[i]<<'x'<<suma/monede[i]<<'='<<monede[i]*(suma/monede[i])<<'\n';
            suma = suma % monede[i];
        }
        i++;
    }
    in.close();
    out.close();
    return 0;
}
```

Exemplu


C:\Users\CLASA215B_...
Fișier Editare Căutare Afișare Codificare
Limbaj Setări Unelte Macro Executare
Module Ferestre ?
suma.in
1 3 29
2 5 3 1
3
C:\Users\CLASA215B_N...
Fișier Editare Căutare Afișare Codificare
Limbaj Setări Unelte Macro Executare Module
Ferestre ?
suma.out
1 ----------
2 5x5=25
3 ----------
4 3x1=3
5 ----------
6 1x1=1
Ln : Windows (CR LF) UTF-8 INS

Rezolvare 2

Concluzie

O metodă *euristică greedy* presupune folosirea unui algoritm *greedy* care nu determină întotdeauna soluția optimă a unei probleme. Dar există situații, în special în practică, unde este mulțumitoare şi o soluție aproximativă a unei probleme. De cele mai multe ori ea se obține repede şi se implementează uşor. O soluție exactă care s-ar obține cu metoda backtracking, practic, este imposibil de obținut, din cauza timpului de execuție exponențial.

Resurse webografice utilizate:

- https://www.hackerearth.com/practice/algorithms/greedy/basics-of-greedy-algorithms/tutorial/