Laborator 9 PA

Metoda Greedy

Minimizarea timpului mediu de așteptare ***[Curs 9, pag 3]***

Din fișierul "tis.txt" se citesc numere naturale nenule reprezentând timpii necesari pentru servirea fiecărei persoane care așteaptă la o coadă.

Să se determine ordinea în care ar trebui servite persoanele de la coadă astfel încât timpul mediu de așteptare să fie minim.

Indicație de rezolvare:

- Se creează o listă de tupluri care să conțină, pentru fiecare persoană, numărul său de ordine în lista inițială și timpul individual de servire pentru acea persoană.
- Se definește o funcție "afisare_timpi_servire (tis)" care primește ca parametru o listă de tupluri de tipul indicat mai sus și afișează pentru fiecare persoană, pe 3 coloane, următoarele informații: numărul de ordine al persoanei, timpul individual de servire și timpul său de așteptare. La final se afișează timpul mediu de așteptare al tuturor persoanelor.
- Se apelează funcția de mai sus pentru lista inițială, iar apoi pentru lista care a fost sortată *crescător după timpul individual de servire*.

Exemplu: Dacă "tis.txt" conține numerele **7 15 3 7 8 3 2 10 5 4 2,**
atunci timpul mediu de așteptare inițial va fi: **41.73**
iar timpul mediu de așteptare după sortarea listei va fi: **24.82**

2. Planificarea optimă a unor spectacole într-o singură sală ***[Curs 10, pag 1]***

Fișierul "spectacole.txt" conține, pe câte un rând, ora de început, ora de sfârșit și numele câte unui spectacol. Să se creeze o listă care să conțină, în tupluri formate din câte 3 șiruri de caractere, cele 3 informații despre fiecare spectacol.

Să se programeze într-o singură sală un număr maxim de spectacole care să nu se suprapună. În fișierul "programare.txt" să se afișeze, pe câte un rând, intervalul de desfășurare și numele pentru spectacolele selectate.

Indicație de rezolvare:

- Se sortează lista de spectacole *crescător după ora de sfârșit*.
- Parcurgând lista sortată, se alege câte un spectacol astfel încât să nu se suprapună cu ultimul spectacol ales (ora de început să fie mai mare sau egală decât ora de sfârșit a ultimului ales).

Exemplu:

spectacole.txt

10:00-11:20 Scufita Rosie
09:30-12:10 Punguta cu doi bani
08:20-09:50 Vrajitorul din Oz
11:30-14:00 Capra cu trei iezi
12:10-13:10 Micul Print
14:00-16:00 Povestea porcului
15:00-15:30 Frumoasa din padurea adormita

programare.txt

08:20-09:50 Vrajitorul din Oz
10:00-11:20 Scufita Rosie
12:10-13:10 Micul Print
15:00-15:30 Frumoasa din padurea adormita

3. Turn de înălțime maximă format din cuburi

Se dă o mulțime de n cuburi. Fiecare cub este caracterizat prin lungimea laturii și culoare. Nu există două cuburi având aceeași dimensiune. Fișierul "cuburi.txt" conține pe prima linie un număr natural nenul n (numărul de cuburi), apoi pe următoarele n linii câte un număr natural nenul (lungimea laturii cubului) și un șir de caractere (culoarea cubului).

Să se construiască un turn de înălțime maximă astfel încât peste un cub cu latura L și culoarea K se poate așeza doar un cub cu latura mai mică strict decât L și culoare diferită de K. În fișierul "turn.txt" să se afișeze componența turnului de la bază spre vârf, pe câte un rând latura și culoarea cubului, apoi la final să se afișeze înălțimea totală a turnului.

Indicație de rezolvare:

- Se sortează lista de cuburi *descrescător după lungimea laturii*.
- La baza turnului se așază cubul de latură maximă, iar apoi se parcurge lista sortată și se adaugă la turn câte un cub care are culoare diferită de ultimul cub adăugat.

Exemplu:

cuburi.txt

6
7 verde
10 rosu
6 albastru
12 rosu
4 rosu
9 verde

turn.txt

12 rosu
9 verde
6 albastru
4 rosu

Inaltime totala: 31

4. Plata unei sume folosind un număr minim de bancnote ***[Curs 9, pag 2]***

Fie o mulțime de bancnote $\{B_0, B_1, \ldots, B_n\}$ astfel încât $B_0 = 1$ (avem mereu bancnota unitate, pentru a putea plăti orice sumă) și $B_i | B_j$, $\forall\ 0 \leq i < j \leq n-2$ (cu excepția ultimelor 2 bancnote, toate valorile se divid cu toate valorile din listă mai mici decât ele). De exemplu se consideră bancnotele românești cu valorile $\{1, 5, 10, 50, 100, 200, 500\}$.

Pentru o sumă de bani S, să se determine o modalitate de a plăti suma S folosind un număr minim de bancnote. Fișierul "bani.txt" conține pe prima linie valorile bancnotelor disponibile (se consideră că avem la dispoziție un număr infinit din fiecare bancnotă), iar pe a doua linie valoarea sumei S. În fișierul "plata.txt" să se afișeze ce bancnote cu valori diferite și câte din fiecare valoare s-au folosit pentru a achita suma S.

Indicație de rezolvare:

- Se sortează lista de bancnote în ordine *descrescătoare*.
- Parcurgând lista cât timp suma rămasă de plată nu este nulă, se alege acea bancnotă cu valoarea mai mică sau egală decât suma rămasă de plată și se scade (de numărul maxim posibil de ori) valoarea bancnotei din suma rămasă.

Exemplu:

bani.txt	**plata.txt**
1 5 10 50 100 200 500 173	173 = 100*1 + 50*1 + 10*2 + 1*3

5. La ora de sport sunt n elevi. Profesorul vrea să execute exerciții de gimnastică cu grupe de câte 2 elevi, dar pentru a putea realiza acest lucru trebuie ca valoarea absolută a diferenței dintre înălțimile celor 2 elevi dintr-o grupă să fie strict mai mică decât un număr natural *h*. Scrieți un program Python care citește de la tastatură două numere naturale *n* și *h* (date fiecare pe o linie), precum și înălțimile celor *n* elevi, după care afișează pe ecran, în forma indicată în exemplu, numărul maxim de grupe formate din câte 2 elevi care se pot realiza respectând condiția indicată anterior. Evident, un elev poate să facă parte din cel mult o grupă! Înălțimile tuturor elevilor și diferența *h* sunt exprimate în centimetri.

Intrare (de la tastatură)	Afisare pe ecran
8 10 172 162 190 183 170 188 165 210	3

Explicații – se pot forma 3 perechi de elevi, prima cu elevii de înălțimi (172, 170), a doua (162,165), a treia (183,190). Soluția nu este unică, o altă soluție poate fi de exemplu (172,170), (162,165), (183,188).