

Tema Nr. 8: Mulțimi disjuncte

Timp alocat: 2 ore

Implementare

Se cere implementarea **corectă** și **eficientă** a operațiilor de bază pe **mulțimi disjuncte** (*capitolul 21.1¹*) și a algoritmului lui **Kruskal** (găsirea arborelui de acoperire minimă, *capitolul 23.2¹*) folosind mulțimi disjuncte.

Se cere să folosiți o pădure de arbori pentru reprezentarea mulțimilor disjuncte. Fiecare arbore trebuie extins cu un câmp *rank* (înălțimea arborelui).

Operațiile de bază pe **mulțimi disjuncte** sunt:

- **MAKE_SET** (x)
 - creează o mulțime nouă ce conține elementul x
- **UNION** (x, y)
 - realizează reuniunea dintre mulțimea care îl conține pe x și mulțimea care îl conține pe y
 - euristica *union by rank* ține cont de înălțime celor doi arbori pentru a realiza reuniunea dintre mulțimi
 - pseudocodul poate fi găsit la *capitolul 21.3¹*
- **FIND_SET** (x)
 - caută mulțime în care se află x
 - euristica *path compression* leagă toate elementele de pe ramura cu x la rădăcina arborelui

¹ Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. *Introduction to Algorithms*

Cerințe

1. Implementare corectă a MAKE_SET, UNION și FIND_SET (5p)

Corectitudinea algoritmilor va trebui demonstrată pe date de intrare de dimensiuni mici

- creați (MAKE) 10 mulțimi + afișare conținuturilor seturilor
- executați secvența UNION și FIND_SET pentru 5 elemente + afișare conținuturilor seturilor

2. Implementarea corectă și eficientă a algoritmului lui Kruskal (2p)

Corectitudinea algoritmului va trebui demonstrată pe date de intrare de dimensiuni mici

- creați un graf cu 5 noduri și 9 muchii + **afișare muchii**
- aplicarea algoritmului lui Kruskal + **afișarea muchiilor alese**

3. Evaluarea operațiilor pe mulțimi disjuncte (MAKE, UNION, FIND) folosind algoritmului lui Kruskal (3p)

! Înainte de a începe să lucrați pe partea de evaluare, asigurați-vă că aveți un **algoritm corect**!

O dată ce sunteți siguri că algoritmul funcționează corect:

- variați n de la 100 la 10000 cu un pas de 100
- pentru fiecare n
 - construiți un graf **conex, neorientat** și **aleatoriu** cu ponderi pe muchii (n noduri, $n*4$ muchii)
 - determinați arborele de acoperire minimă folosind algoritmul lui Kruskal
 - evaluați efortul computațional al fiecărei operații de bază (MAKE, UNION, FIND – *reprezentați rezultatele sub forma unui grafic cu trei serii*) pe mulțimi disjuncte ca suma comparațiilor și atribuțiilor efectuate; astfel, ar trebui să existe **3 serii în grafic**, câte una pentru fiecare operație.