Tema 2 – Arbori

Ana-Maria Simion, Andreea Duțulescu, Anca Bălușoiu, Cătălin Rîpanu,
Alexandru Axenia, Mihai Nan

Data postării: 30.04.2025
Deadline: 21.05.2025 ora 23:59

1 Descriere

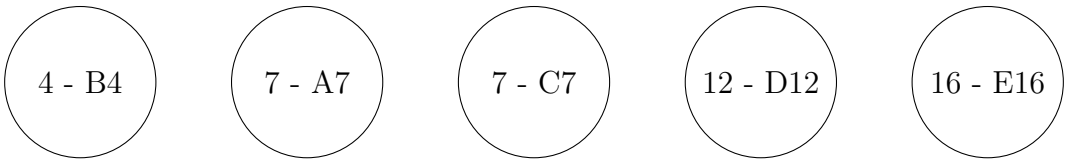
Agenția Spațială Română și-a propus să dezvolte propria rețea de sateliți pentru a observa și explora spațiul. Rețeaua de sateliți trebuie realizată sub formă de arbore binar. Sateliții se conectează între ei, și există un satelit rădăcină, aproape de atmosfera Pământului, care face legătura cu toți ceilalți sateliți.

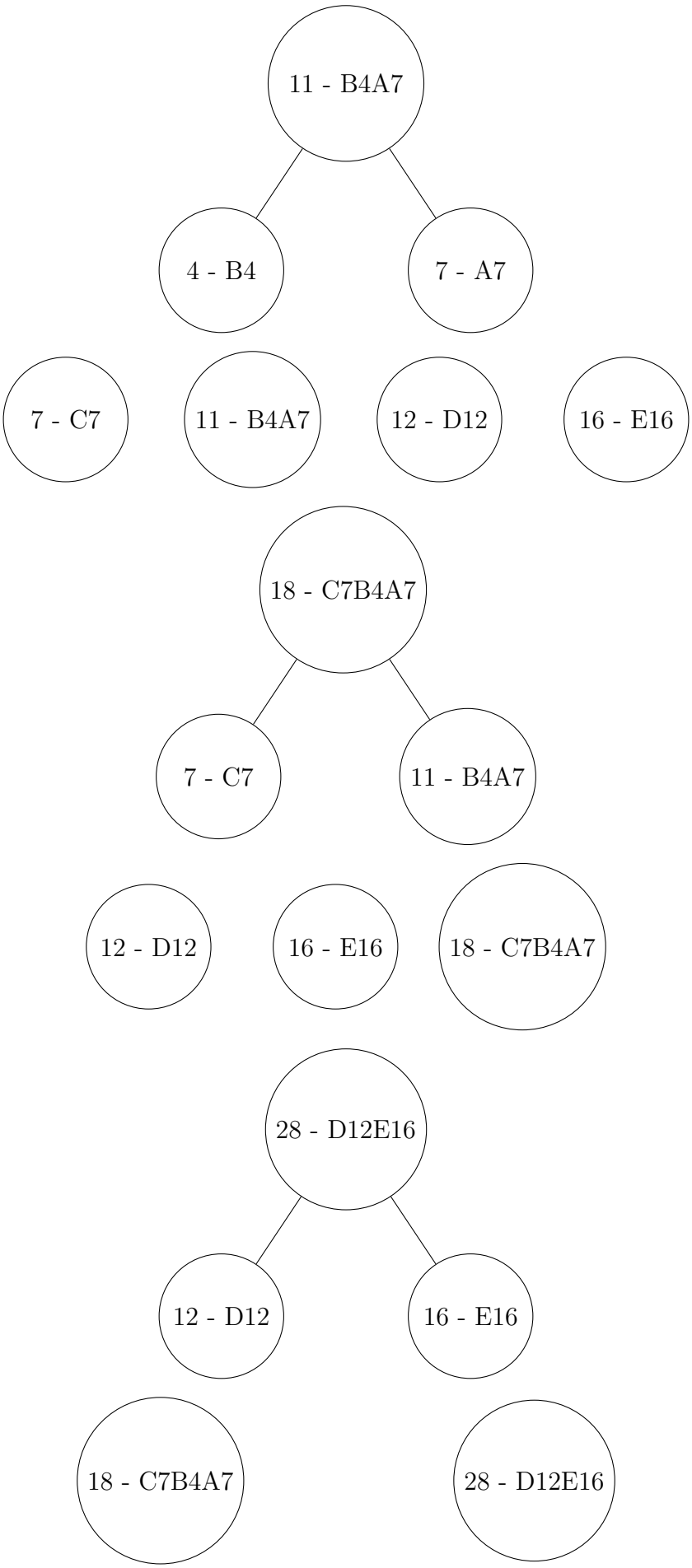
Fiecare satelit ce va urma să fie în rețea are o frecvență cu care raportează datele. Astfel, pentru a facilita comunicarea, ne dorim ca sateliții care raportează mai des să fie mai aproape de Pământ, iar sateliții care raportează mai rar să se afle la o distanță mai mare. Orice legătură între doi sateliți are o distanță constantă K. Pentru realizarea conexiunii sateliților, se pot folosi oricâți sateliți de legătură va fi necesar. Sateliții de legătură sunt meniți să asigure comunicarea între sateliții principali.

Pentru a asigura conexiunea între sateliți, se adaugă sateliți de legătură ori de câte ori este necesar. Acești sateliți intermediari sunt responsabili pentru menținerea structurii arborelui binar. Legarea sateliților în rețea se face astfel:

1. Se identifică cei doi sateliți cu cele mai mici frecvențe de raportare.
2. Se introduce un satelit de legătură ca nod părinte pentru acești doi sateliți. Frecvența de raportare a satelitului de legătură este calculată ca suma frecvențelor celor doi sateliți copil. Numele acestuia este concatenarea dintre numele fiului stang și numele fiului drept.
3. Satelitul de legătură este apoi adăugat in Min-Heap, repetând procesul până când toți sateliții sunt legați într-un singur arbore binar.

Exemplul 1




11 - B4A7
4 - B4
7 - A7
7 - C7
11 - B4A7
12 - D12
16 - E16
18 - C7B4A7
7 - C7
11 - B4A7
12 - D12
16 - E16
18 - C7B4A7
28 - D12E16
12 - D12
16 - E16
18 - C7B4A7
28 - D12E16

1.1 Cerința 1

Primim ca input o listă de sateliți, fiecare caracterizat printr-un identificator unic și o frecvență de raportare a datelor. Să se implementeze o funcție care construiește arborele binar, astfel încât să se respecte criteriul de organizare: sateliții cu frecvență mai mare sunt mai aproape de rădăcină, iar cei cu frecvență mai mică sunt mai departe. Funcția va returna rădăcina rețelei de sateliți. Pentru construirea acestei rețele de tip arbore se va folosi o structură de Min-Heap pentru a extrage, la fiecare pas, sateliții cu cele mai mici valori de frecvență de raportare. Satelitul cu frecvența de raportare mai mică va fi fiul stâng al nodului de legătură. Dacă ambii sateliți au aceeași frecvență de raportare, în partea stângă va fi plasat satelitul al cărui nume apare primul în ordine alfabetică. Dacă există mai mulți sateliți cu aceeași frecvență de raportare, extragerea se va face în ordine alfabetică. Să se afișeze arborele rezultat pe niveluri. Informația din nodurile de legătură reprezintă suma timpilor de raportare pentru sateliții din cei 2 subarbori, iar numele este obținut prin concatenarea numelor celor 2 noduri copil.

Format Input:

```
Numărul de sateliți
Frecvența_de_raportare_1 Numele_satelitului_1
Frecvența_de_raportare_2 Numele_satelitului_2
...
Frecvența_de_raportare_N Numele_satelitului_N
```

Restricții pentru datele de intrare: Frecvența de raportare este un număr întreg. Numele unui satelit este un șir de cel mult 15 caractere.

Input:

```
5
4 B4
7 A7
7 C7
12 D12
16 E16
```

Output:

```
46-C7B4A7D12E16
18-C7B4A7 28-D12E16
7-C7 11-B4A7 12-D12 16-E16
4-B4 7-A7
```

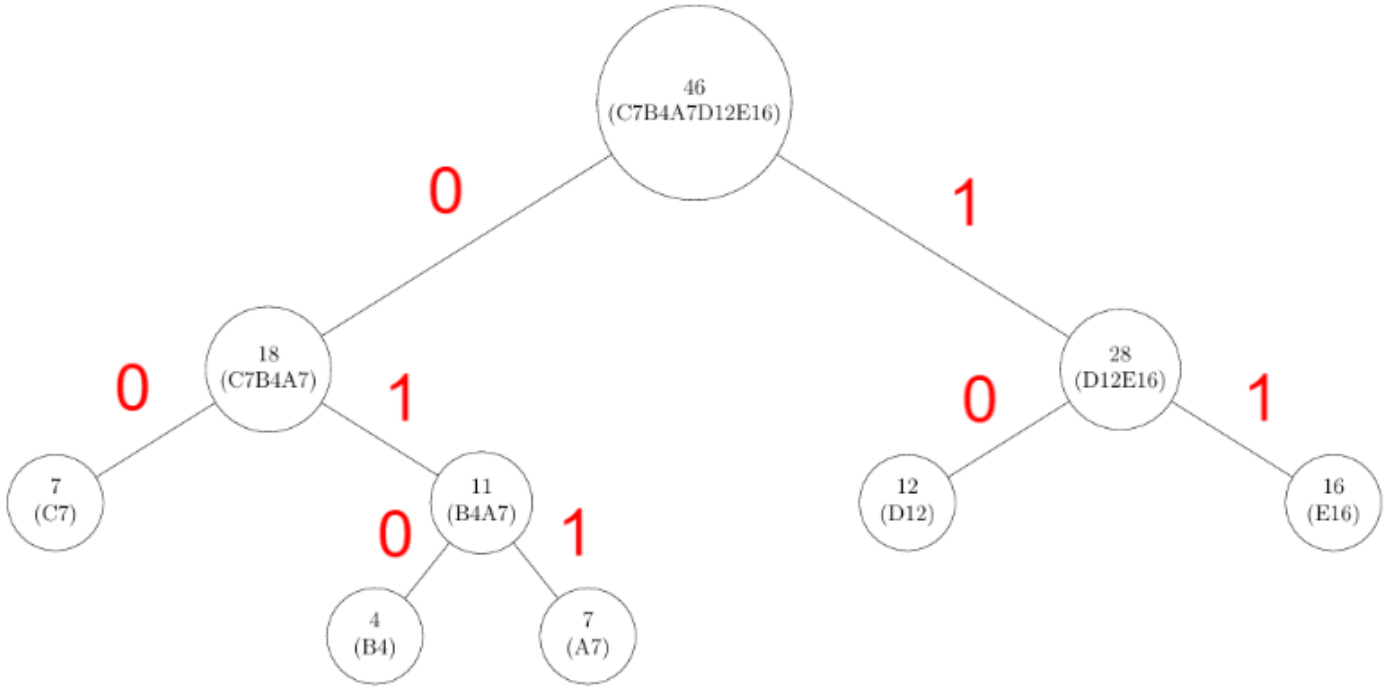


Figure 1: Arborele final pentru Exemplul 1

1.2 Cerința 2

Agenția a recepționat un mesaj de la o sursă necunoscută, în codificare binară, și și-au dat seama că fiecare 0 reprezintă o mișcare la stânga în rețeaua de sateliți, iar fiecare valoare de 1 reprezintă o mișcare la dreapta. Astfel, și-au dat seama că trebuie să descifreze codificarea pentru a determina în ce ordine să inspecteze informațiile colectate de sateliți. Primim ca input o secvență de valori 1 și 0 și trebuie să determinăm sateliții codificați în secvență.

Format Input:

```
Numărul de codificări
Codificarea1
Codificarea2
....
CodificareaN
```

Restricții pentru datele de intrare:

Codificarea este un șir de cel mult 1000 caractere.

Input:

```
2
00010
1110011
```

Output:

```
C7 B4
E16 D12 A7
```

Explicație:

```
00|010
11|10|011
```

1.3 Cerința 3

Pentru a răspunde la mesaj, au distribuit diferite bucăți de informație în anumiți sateliți, iar codificarea trebuie trimisă în același fel în care a fost primită. Se va citi un număr N de sateliți și apoi numele acestora. Trebuie să se determine codificarea sateliților în ordinea primită.

Input:

```
3
A7
D12
E16
```

Output:

```
0111011
```

1.4 Cerința 4

Câteodată apar erori de comunicare între anumiți sateliți și atunci trebuie să determinăm satelitul părinte care face legătura între nodurile respective. Primim ca input o listă cu indicatorii unici

ai sateliților care au erori de transmitere de date și trebuie să determinăm cel mai apropiat satelit părinte comun pentru toate acele noduri, pentru a-l repara. Funcția trebuie să întoarcă identificatorul unic al satelitului părinte.

Format Input:

```
Numărul de sateliți
Numele Satelitului 1
Numele Satelitului 2
...
Numele Satelitului N
```

Input:

```
3
B4
A7
C7
```

Output:

```
C7B4A7
```

1.5 Cerința 5

Proiectul a fost un succes și agenția spațială a hotărât să extindă și mai mult rețeaua principală în următorul mod: Fiecare nod din rețeaua principală, mai puțin radăcina, poate să fie legat de nodul rădăcină al unui arbore multicăi. Nodurile din arborele principal stochează acum și o legătură către rădăcina arborelui multicăi. Nodul din arborele multicăi conține, la fel ca celelalte noduri din arborele principal, o valoare pentru frecvența de raportare și un nume. Citirea arborelui multicăi se realizează pe nivel.

Să se implementeze o funcție care adaugă aceste noduri în rețeaua principală și o funcție care determină distanța între două noduri date (oricare dintre nodurile date pot fi din arborele principal sau din arborele multicăi). Fiecare legătură dintre noduri reprezintă o unitate de măsură. În figura 2 sunt exemplificate toate cazurile posibile: ambele noduri sunt din arborele principal, un nod este din arborele principal și celălalt este dintr-un arbore multicăi, ambele noduri sunt într-un arbore multicăi sau cele două noduri sunt în arbori multicăi diferiți. Distanța între nodurile albastre este 2. Distanța între nodurile galbene este 4. Distanța între nodurile roșii este 11. Distanța între nodurile verzi este 5.

Format Input:

```
Numărul de arbori ce vor fi adăugați
Nume nod din arborele principal de care se leagă rădăcina arborelui
multicăi
Rădăcina arborelui multicăi
Numărul noduri părinte din arborelui multicăi
Nod părinte
Numărul noduri copil
Frecvență_raportare_nod_copil Nume_nod_copil
Nod1 Nod2
```

unde `Nod1` și `Nod2` sunt cele două noduri pentru care trebuie să determinăm distanța.

Input:

```
2
B4
12 Y12
3
Y12
4
5 Y78
63 Y52
6 YU2
1 Y01
Y78
2
94 Y20
144 Y62
YU2
1
56 Y6
D12
5 X1
2
X1
3
36 X6
7 X2
92 X7
X2
3
8 X3
2 X4
89 X5
Y6 X3
```

Output:

```
11
```

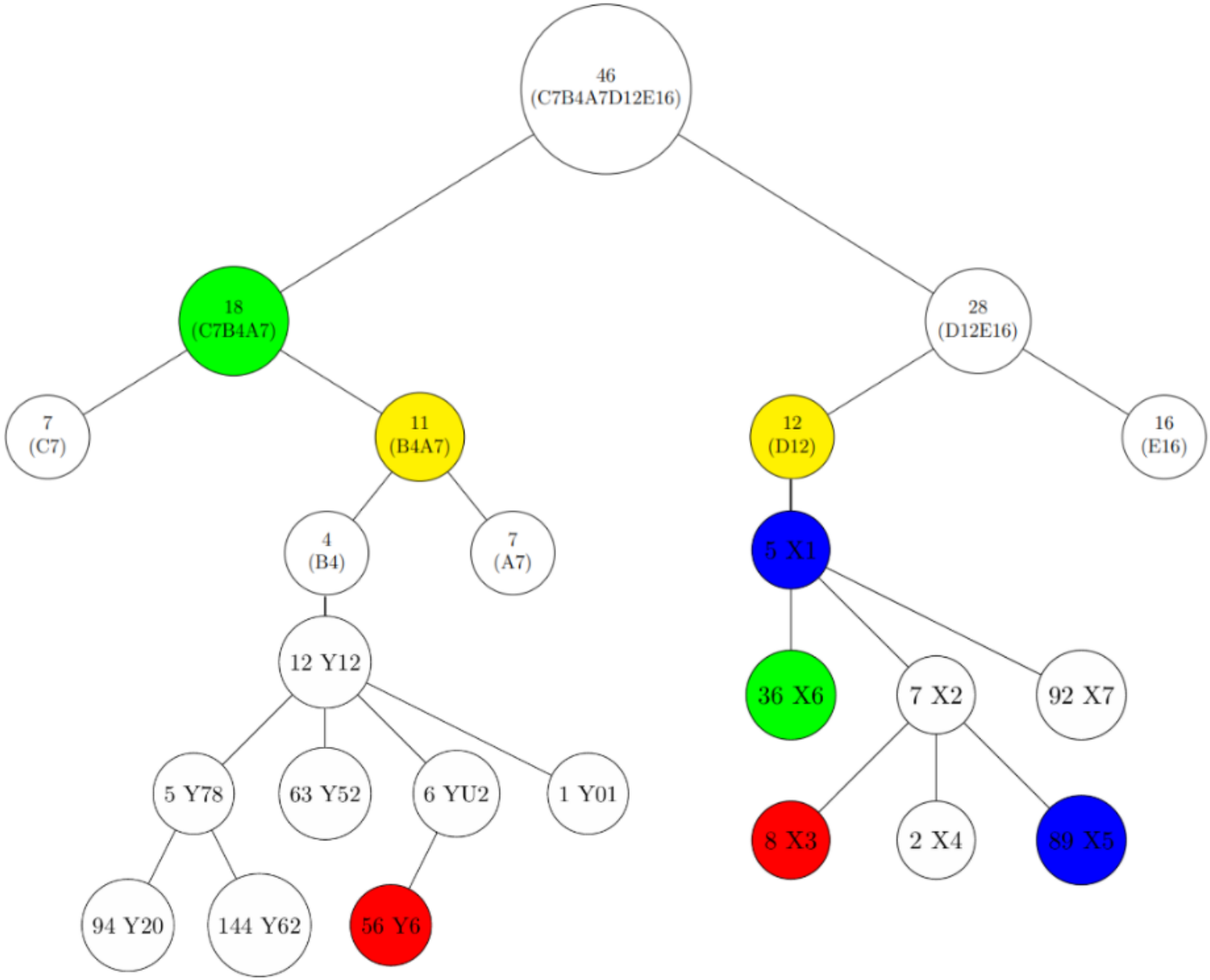


Figure 2: Exemplu de arbore pentru exercițiul 5

2 Instrucțiuni pentru predarea temelor

Temele trebuie să fie încărcate atât pe checker-ul automat, cât și pe Moodle, la secțiunea corespunzătoare. NU se acceptă teme trimise pe e-mail sau prin alt mijloc decât Moodle.

O rezolvare constă într-o arhivă de tip .zip care conține toate fișierele sursă, alături de un Makefile, care va fi folosit pentru compilare, și un fișier README, în care se vor preciza detaliile implementării. Makefile-ul trebuie să conțină obligatoriu regulile build și clean. Regula build trebuie să aibă ca efect compilarea surselor și crearea binarului tema2.

Programul vostru va primi, ca argumente în linia de comandă, numele fișierului de intrare și al celui de ieșire, dar și o opțiune, în felul următor:

```
./tema2 [-c1 |-c2 |-c3 |-c4 |-c5] [fișier_intrare] [fișier_ieșire]
```

- c1 indică faptul că programul va rezolva cerința 1;
- c2 indică faptul că programul va rezolva cerința 2;
- c3 indică faptul că programul va rezolva cerința 3;
- c4 indică faptul că programul va rezolva cerința 4;
- c5 indică faptul că programul va rezolva cerința 5;
- fișier_intrare reprezintă numele fișierului de intrare;
- fișier_ieșire reprezintă numele fișierului de ieșire, în care se va scrie, în funcție de comanda primită, rezultatul execuției programului.

3 Punctaj

O temă cu toate cerințele de bază rezolvate corect valorează 100 de puncte. 95 de puncte se vor acorda pentru teste și 5 puncte pentru README. În urma corectării manuale, punctajul acordat de checker-ul automat poate fi scăzut cu maxim 15 puncte pentru coding style. De asemenea, conținutul fișierului README va fi verificat manual, iar punctajul obținut pentru README poate fi diminuat. Punctajul pe teste este următorul:

Cerinta	Punctaj
Cerința 1	30 puncte
Cerința 2	20 puncte
Cerința 3	20 puncte
Cerința 4	25 puncte
README	5 puncte
BONUS Cerința 5	20 puncte

Atenție!

Orice rezolvare care nu conține structurile de date specificate **NU** este punctată.
Temele vor fi punctate doar pentru testele care sunt trecute pe vmchecker.
Nu lăsați warning-urile nerezolvate, deoarece veți fi depunctați.
Tema este individuală! Toate soluțiile trimise vor fi verificate, folosind o unealtă pentru detectarea plagiatului.