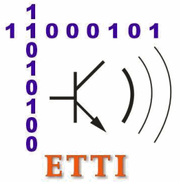
UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI



Laborator 8

Arbori

Liviu-Daniel ȘTEFAN, Mihai DOGARIU, Bogdan IONESCU

Cuprins

Partea I – Noțiuni teoretice

L2.1. Arbori
L2.2. Arbori Binari de Căutare

Partea II – Aplicații Laborator

Rezolvare aplicații Moodle

Arbori

Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Arbori

Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Proprietăți:

- Există un nod și numai unul care se numește rădăcina arborelui;

Arbori

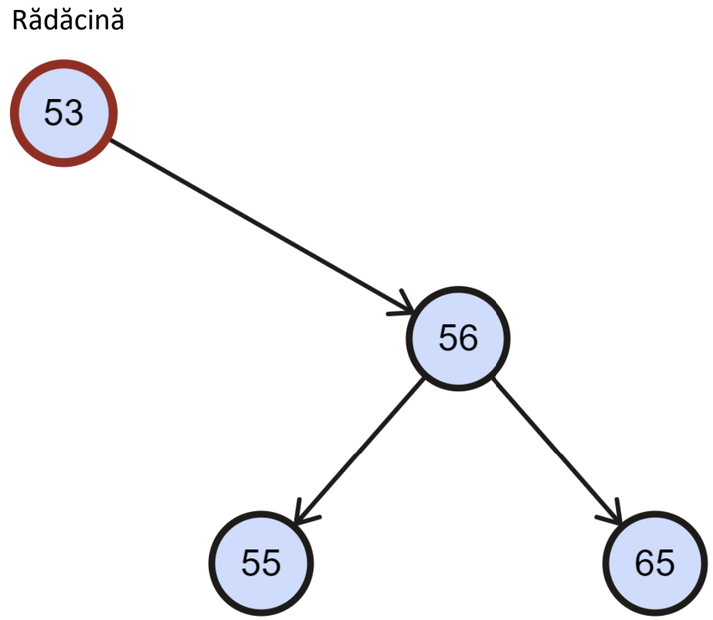
Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Proprietăți:

- Există un nod și numai unul care se numește rădăcina arborelui.
- Celelalte noduri formează submulțimi disjuncte ale lui A, care formeaza câte un arbore. Arborii respectivi se numesc subarbori ai rădăcinii;



Arbori

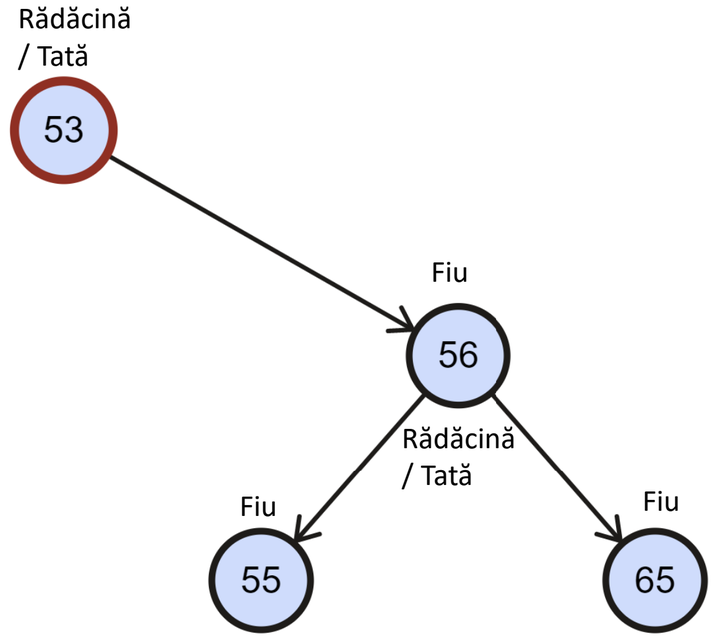
Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Proprietăți:

- Există un nod și numai unul care se numește rădăcina arborelui.
- Celelalte noduri formează submulțimi disjuncte ale lui A, care formeaza câte un arbore. Arborii respectivi se numesc subarbori ai rădăcinii;
- Un nod rădăcină se spune că este un nod tată, iar subarborii rădăcinii sunt descendenții acestuia. Rădăcinile descendenților unui nod tată sunt fiii lui.



Arbori

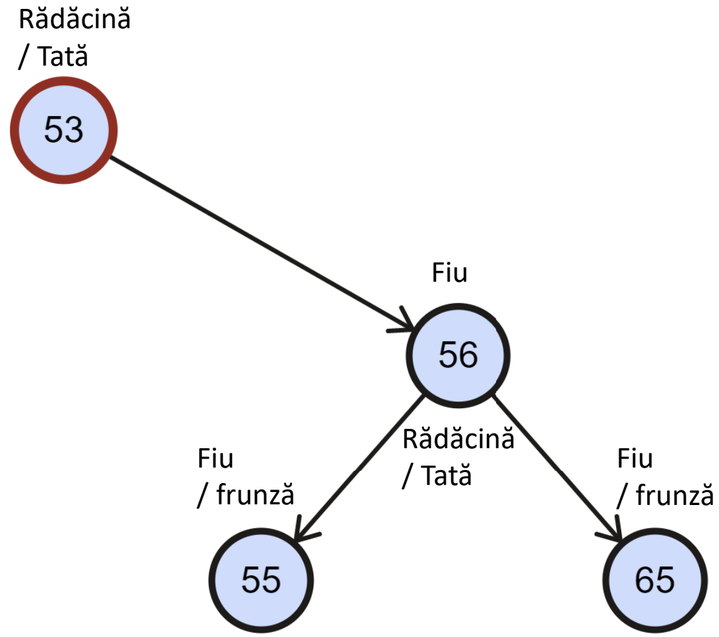
Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Proprietăți:

- Există un nod și numai unul care se numește rădăcina arborelui.
- Celelalte noduri formează submulțimi disjuncte ale lui A, care formeaza câte un arbore. Arborii respectivi se numesc subarbori ai rădăcinii;
- Un nod rădăcină se spune că este un nod tată, iar subarborii rădăcinii sunt descendenții acestuia. Rădăcinile descendenților unui nod tată sunt fiii lui.
- Un nod căruia nu îi corespunde un alt nod se numește nod terminal sau frunză.



Arbori

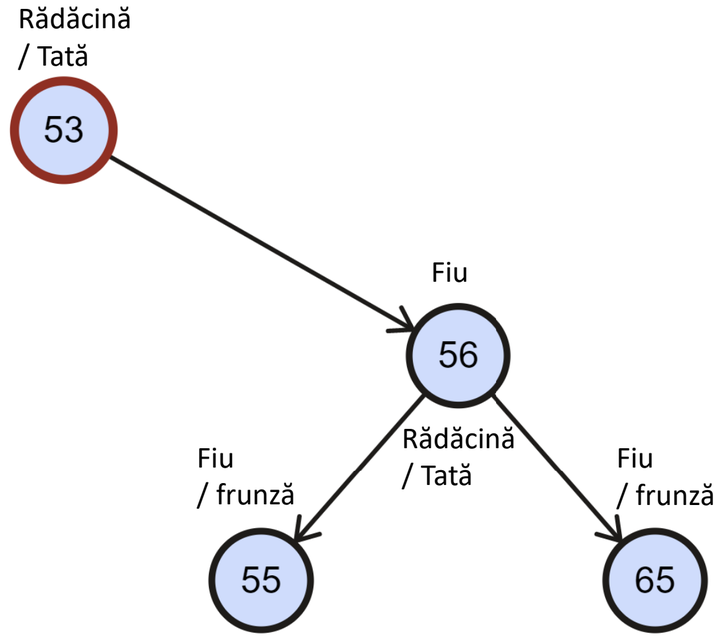
Aspecte teoretice

Arbori: structure de date de natură recursivă și dinamică.

$$A = \{A1, A2, A3, \dots, An\}, \qquad \text{unde n} \geq 0$$

Proprietăți:

- Există un nod și numai unul care se numește rădăcina arborelui.
- Celelalte noduri formează submulțimi disjuncte ale lui A, care formeaza câte un arbore. Arborii respectivi se numesc subarbori ai rădăcinii;
- Un nod rădăcină se spune că este un nod tată, iar subarborii rădăcinii sunt descendenții acestuia. Rădăcinile descendenților unui nod tată sunt fiii lui.
- Un nod căruia nu îi corespunde un alt nod se numește nod terminal sau frunză.
- Nu există cicluri, ceea ce înseamnă că, pornind de la un nod oarecare nu se poate parcurge un anumit traseu, astfel încât să se ajungă înapoi la nodul de plecare.



Arbori Binari de Căutare

Arbore binar de căutare: arbore binar particular care stochează suplimentar o valoare numerică cu rol de ordonare. Pentru oricare nod *n*, fiecare dintre descendenții din subarborele din stânga are valoarea numerică mai mică decât a nodului *n*, iar fiecare din descendenții din subarborele din dreapta va avea valoarea informației mai mare sau egală.

Definiție

```
struct NOD
{
 int x;
 struct NOD *NOD_stanga;
 struct NOD *NOD_dreapta;
};
```

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

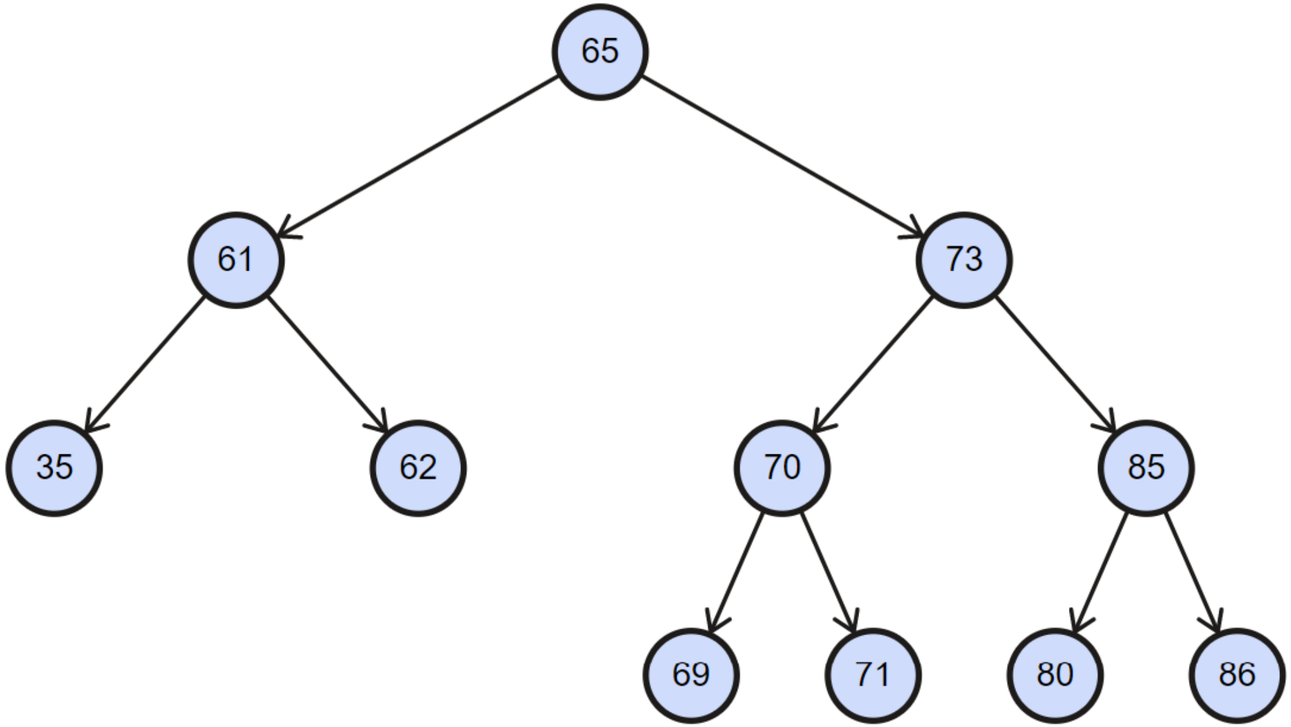
1. **Parcurgere breadth-first search**
2. **Parcurgere depth-first search**

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)

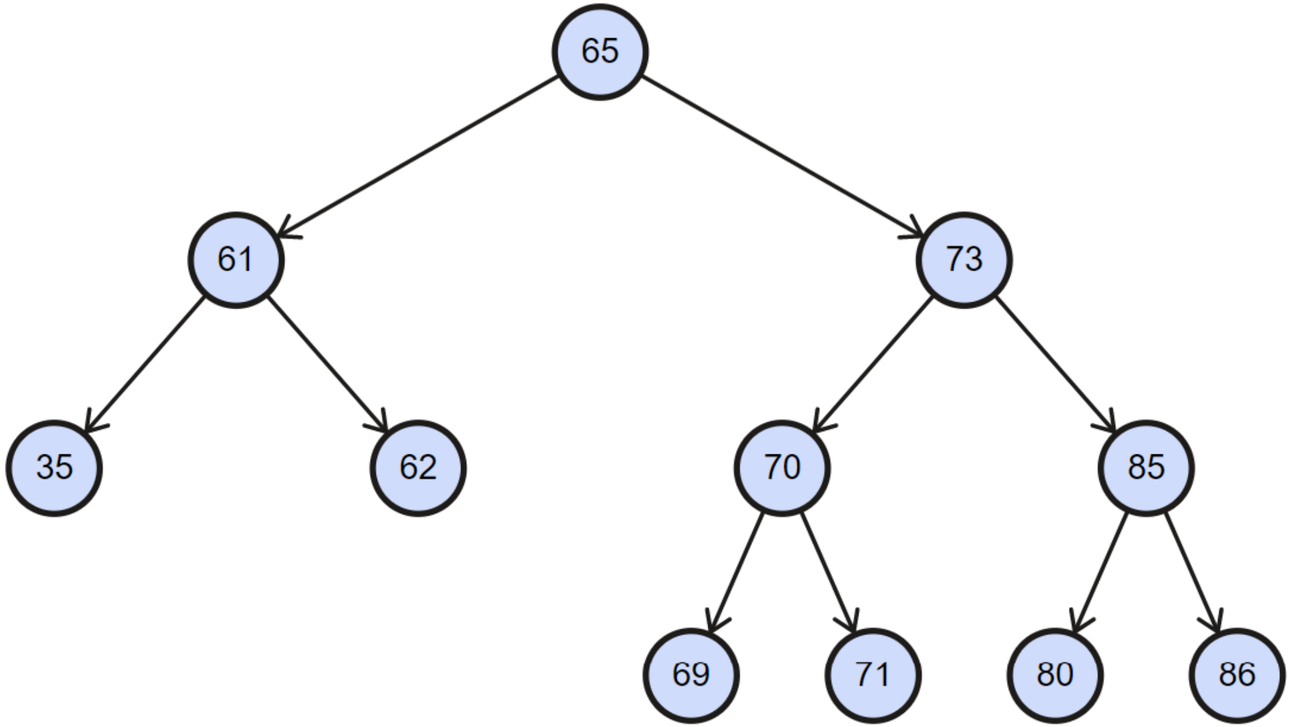


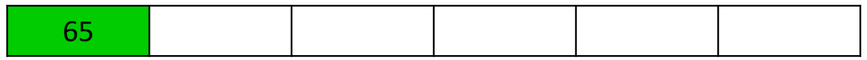
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





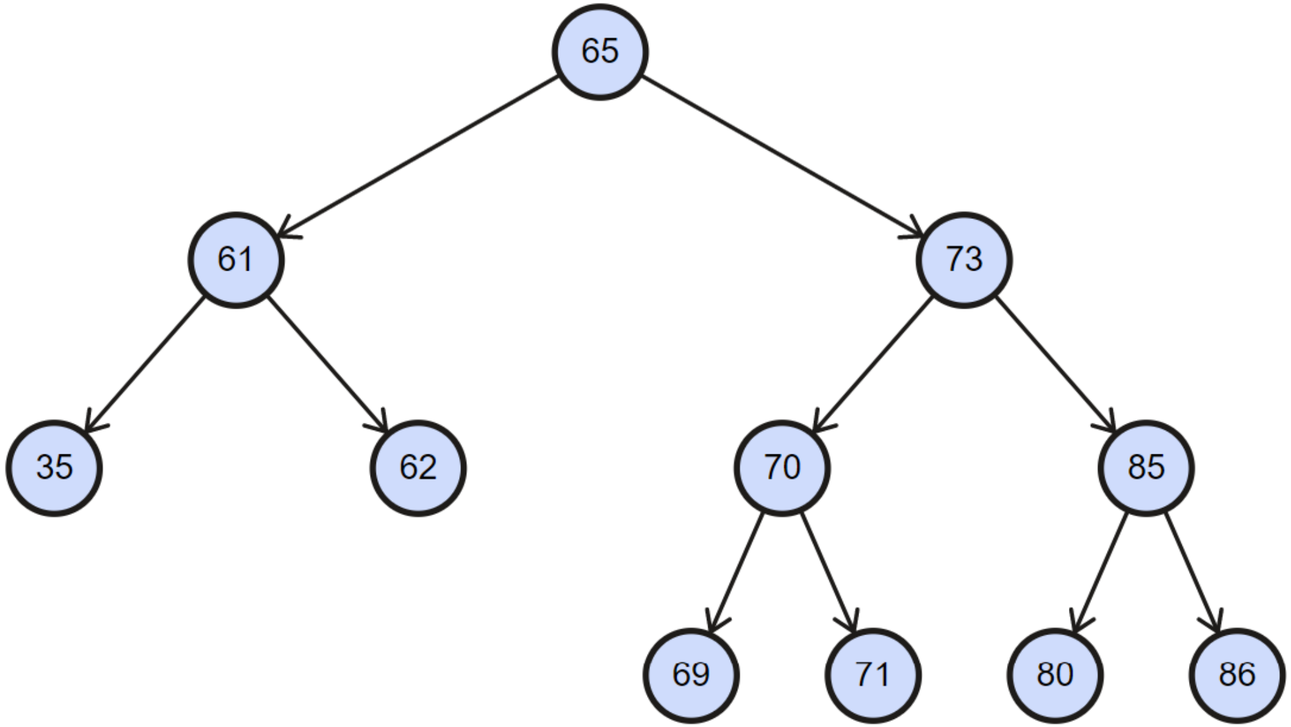
Output:

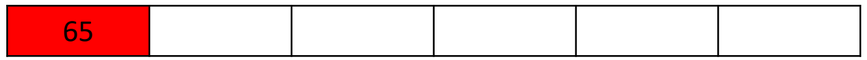
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





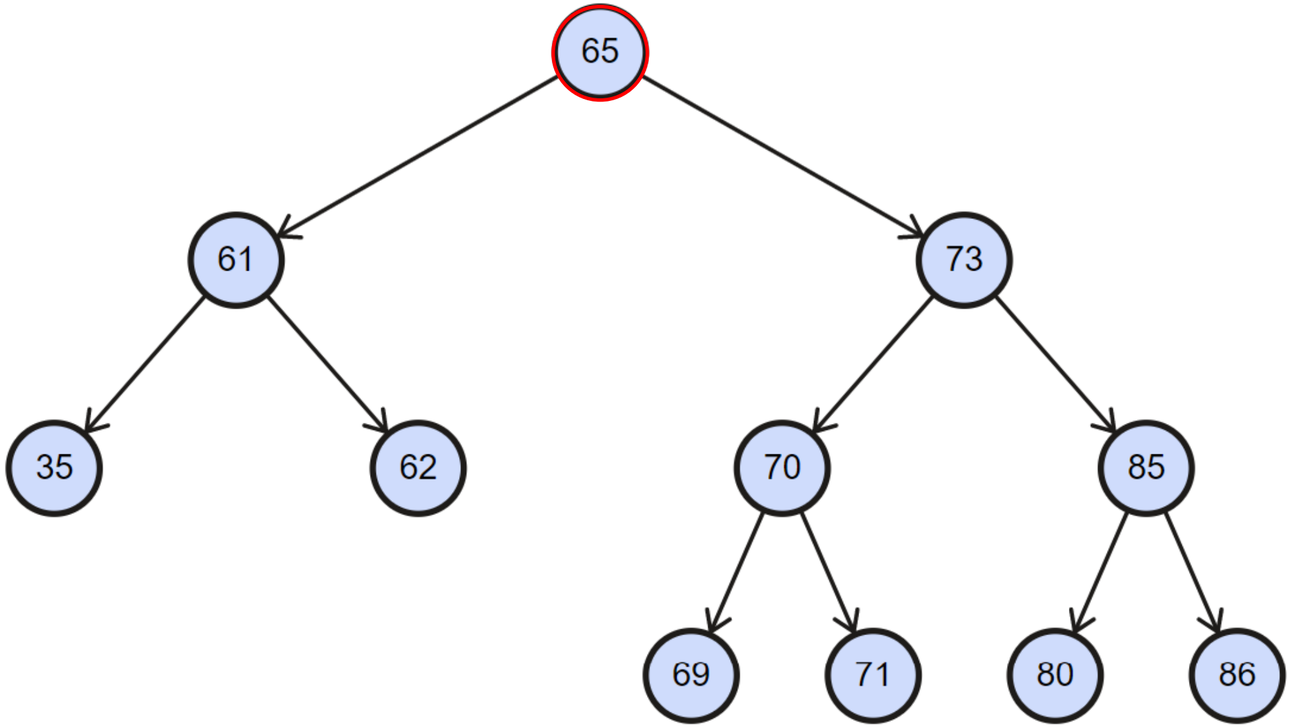
Output:

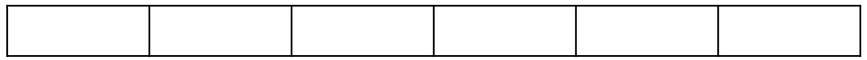
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





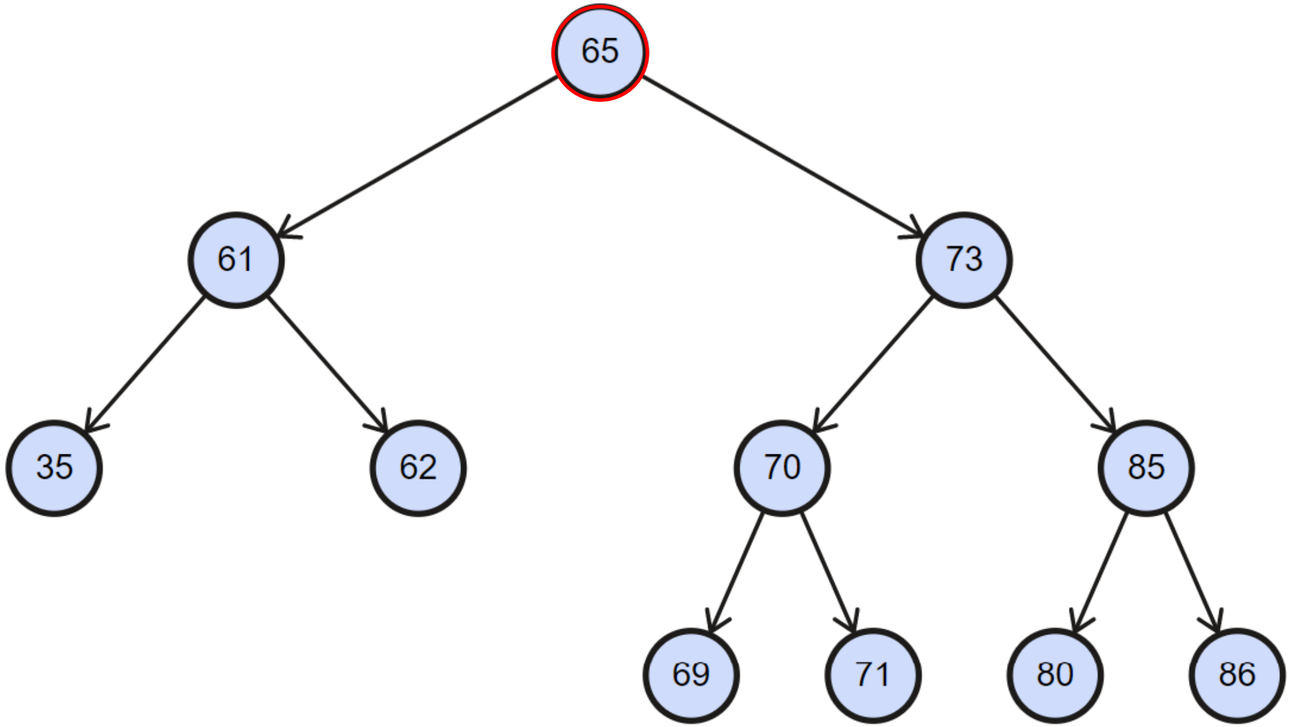
Output: 65

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



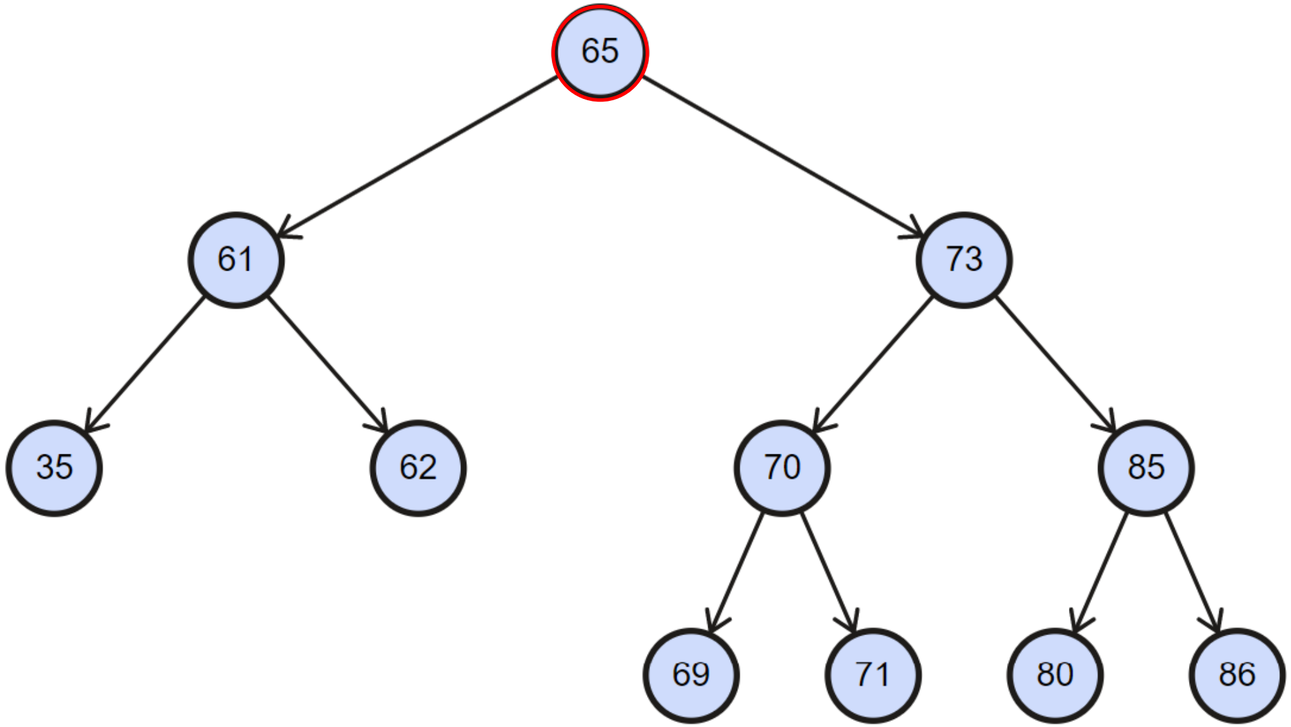
Output: 65

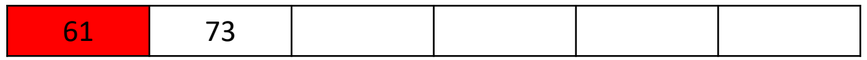
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





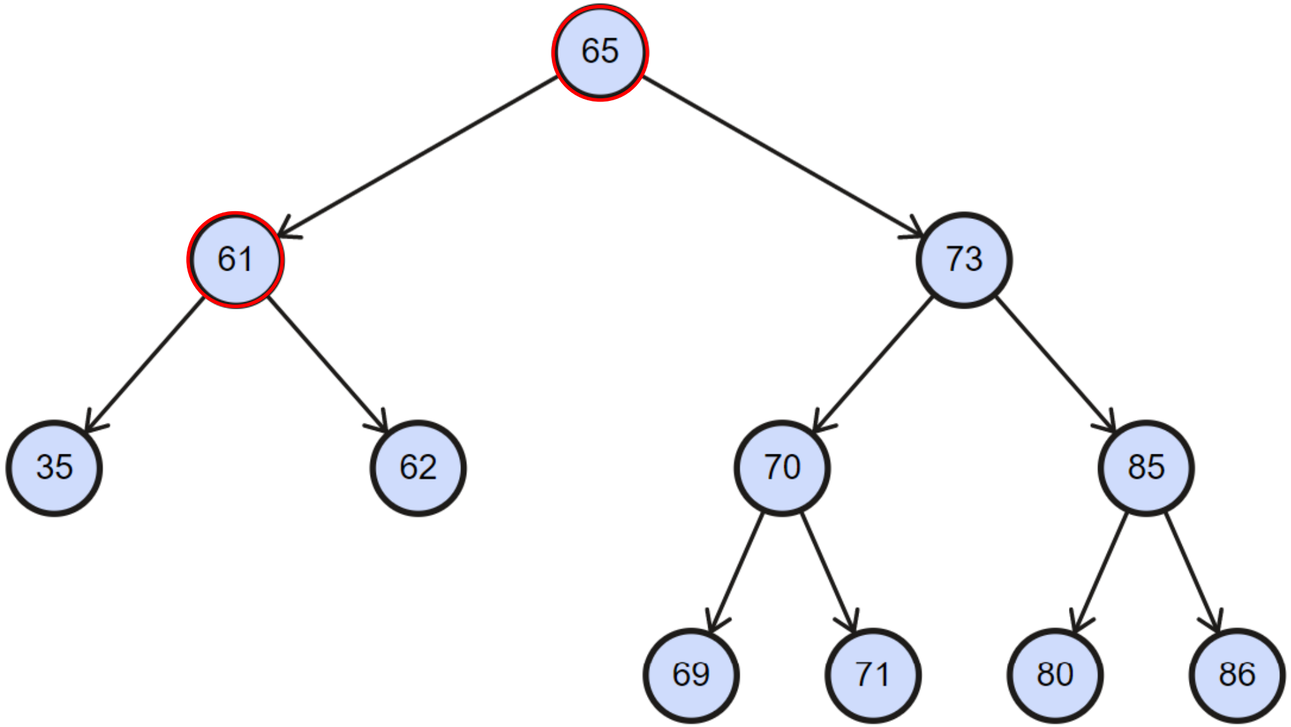
Output: 65

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



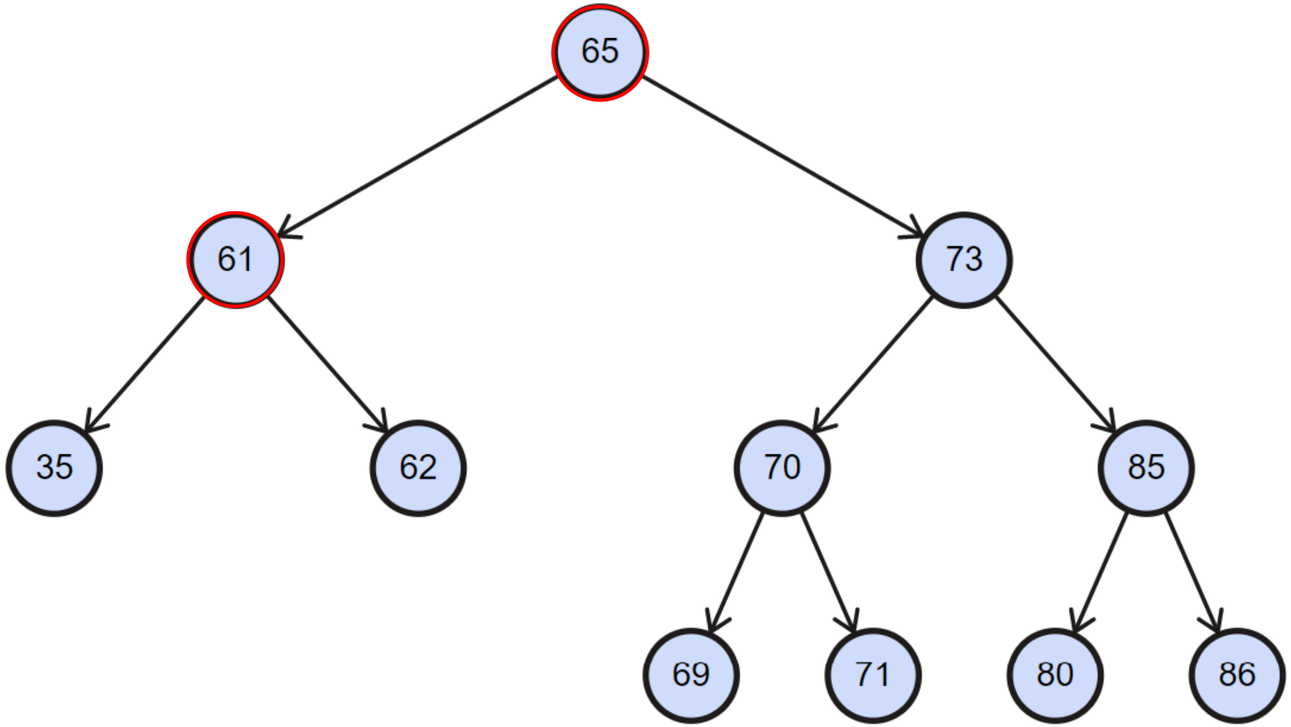
Output: 65, 61

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



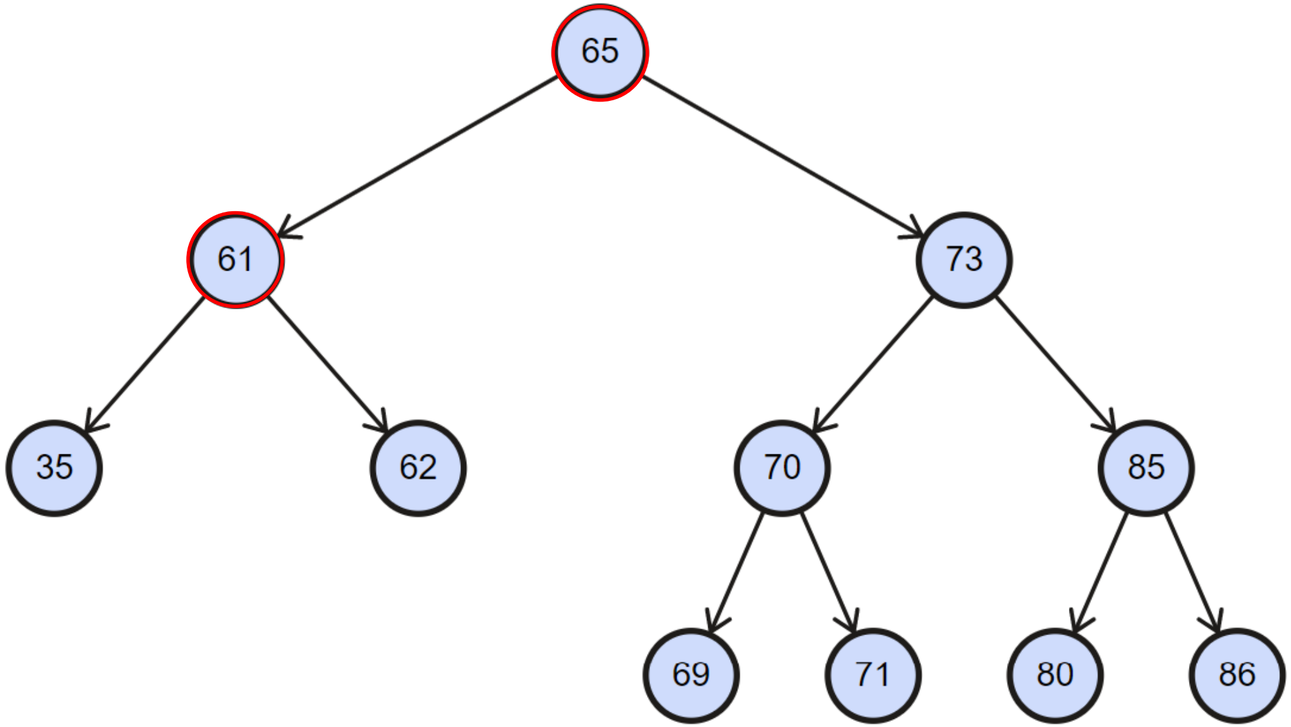
Output: 65, 61

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



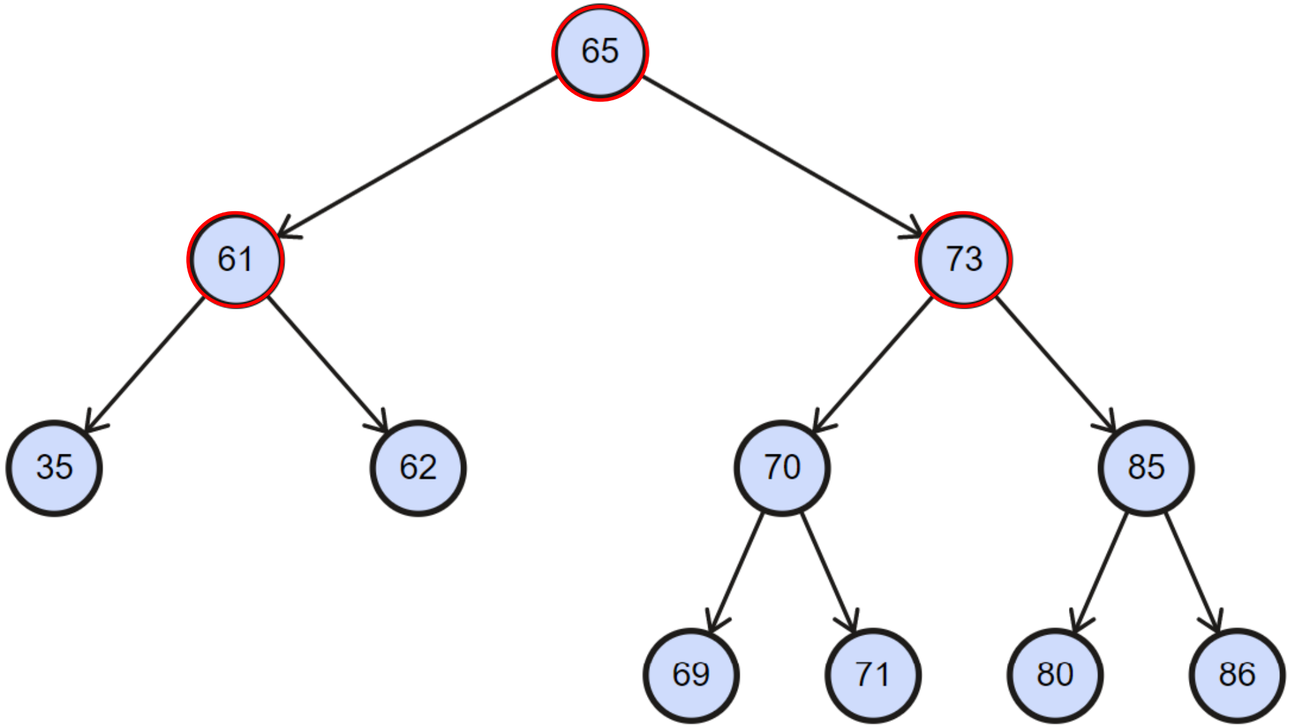
Output: 65, 61

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



35	62				

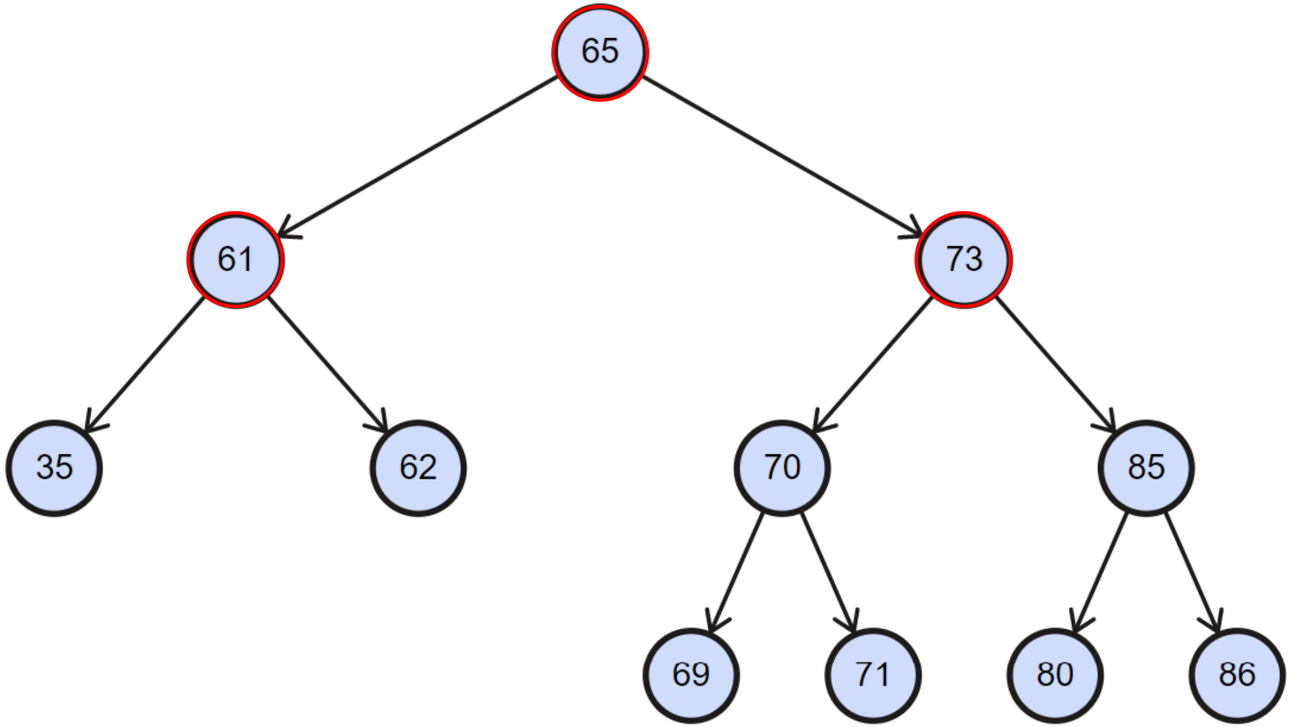
Output: 65, 61, 73

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



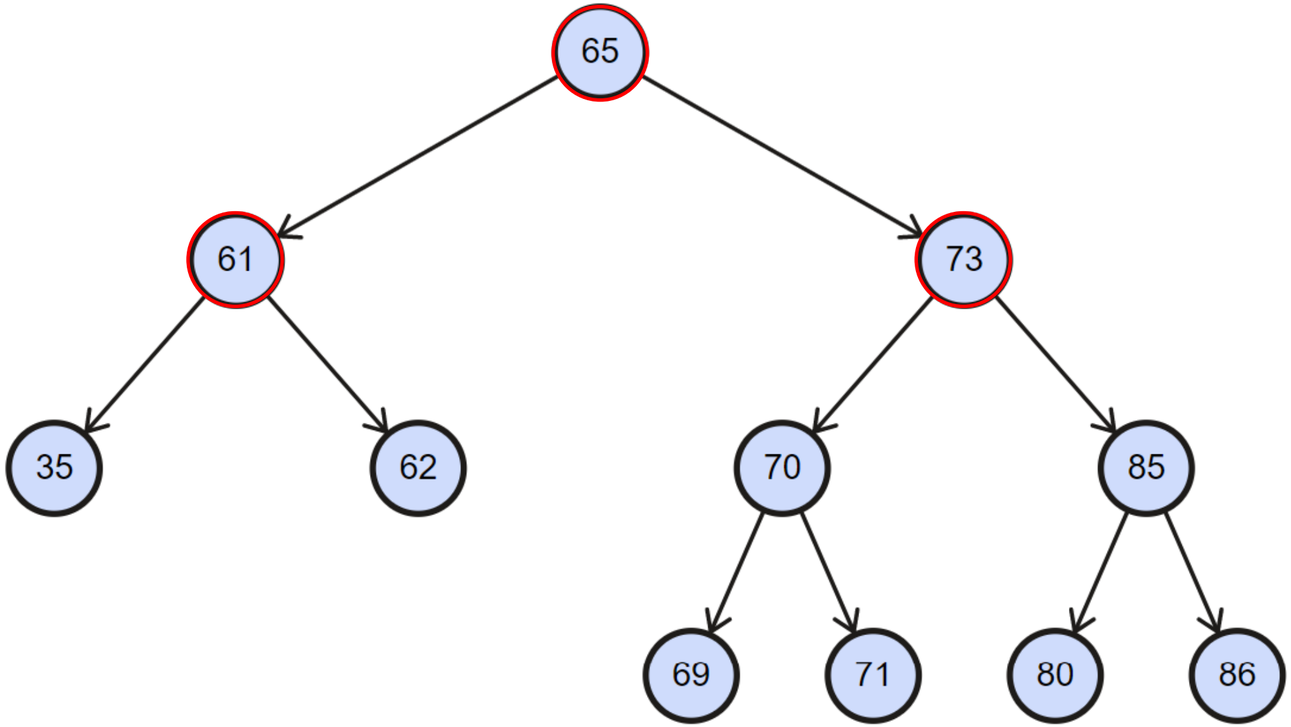
Output: 65, 61, 73

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



35	62	70	85		

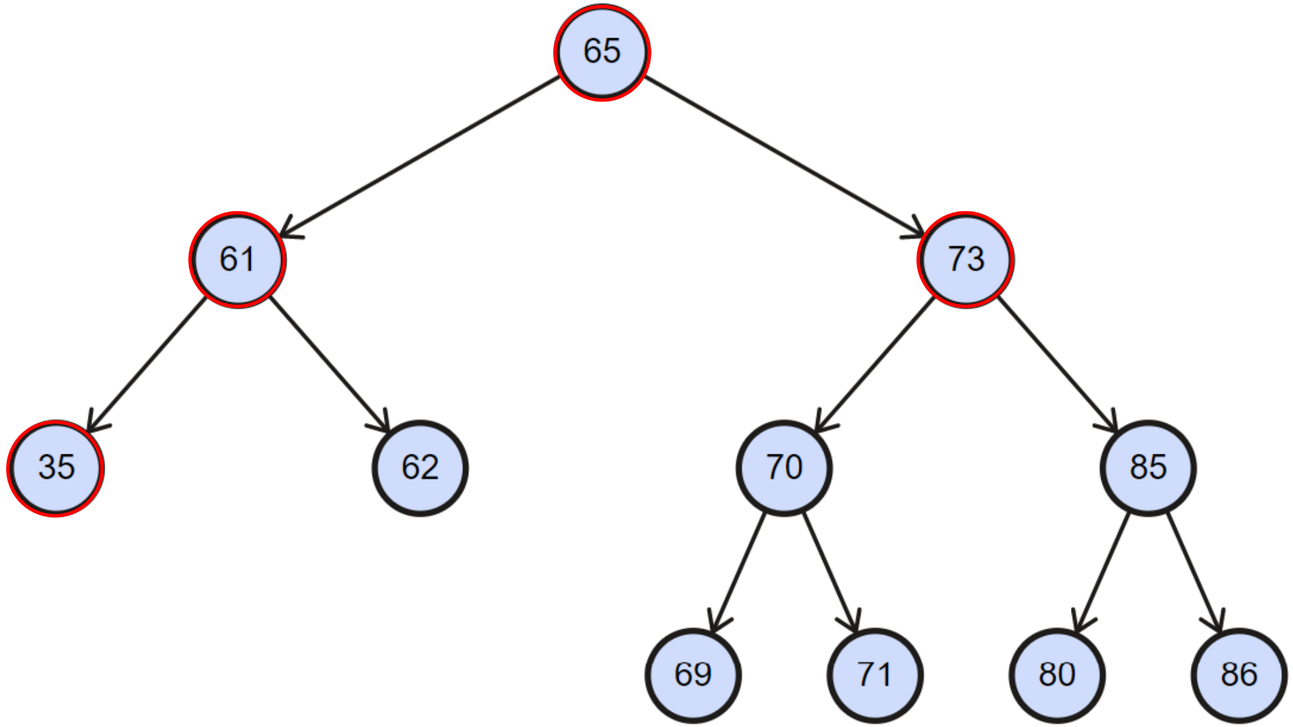
Output: 65, 61, 73

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



62	70	85			

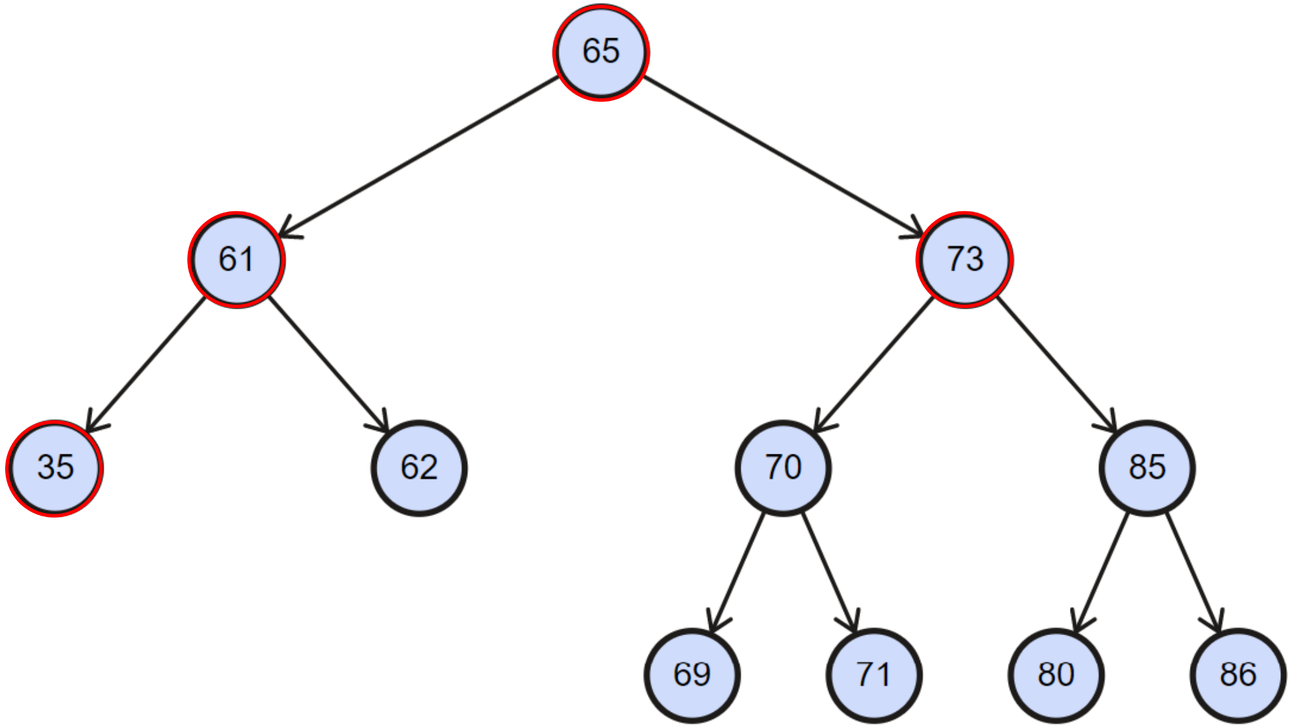
Output: 65, 61, 73, 35

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



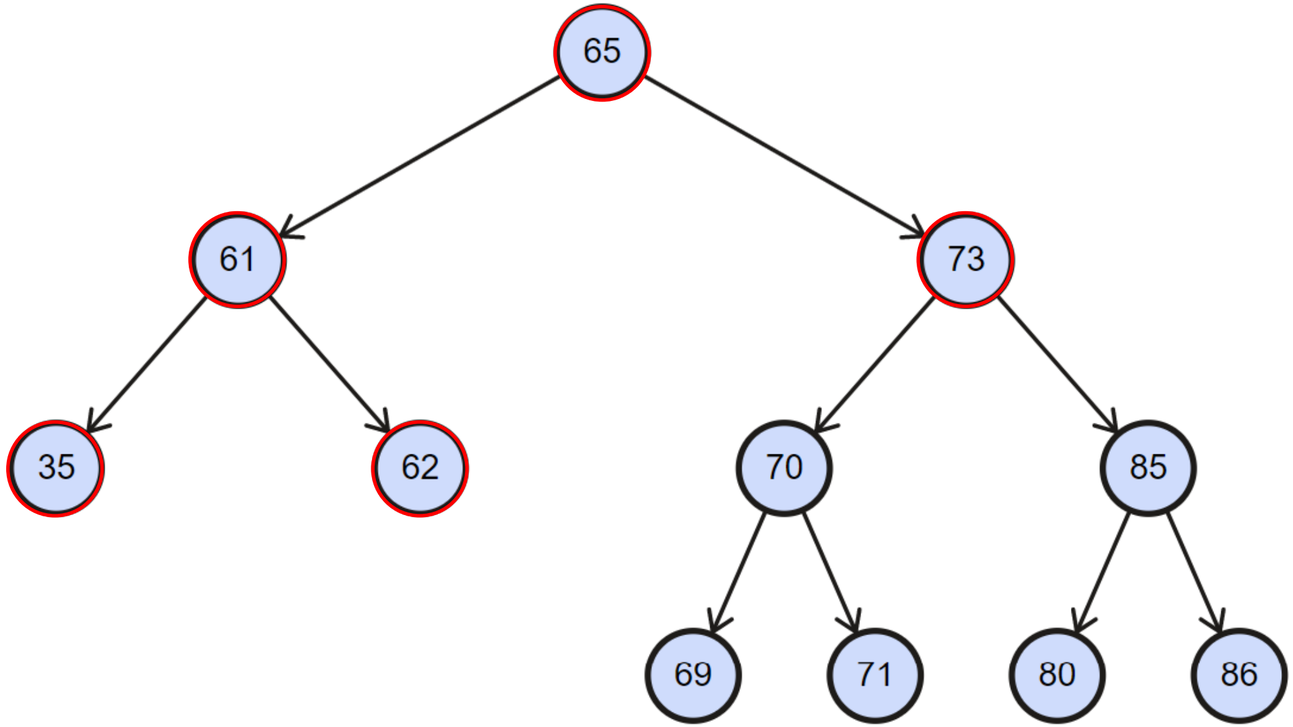
Output: 65, 61, 73, 35

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



70	85				

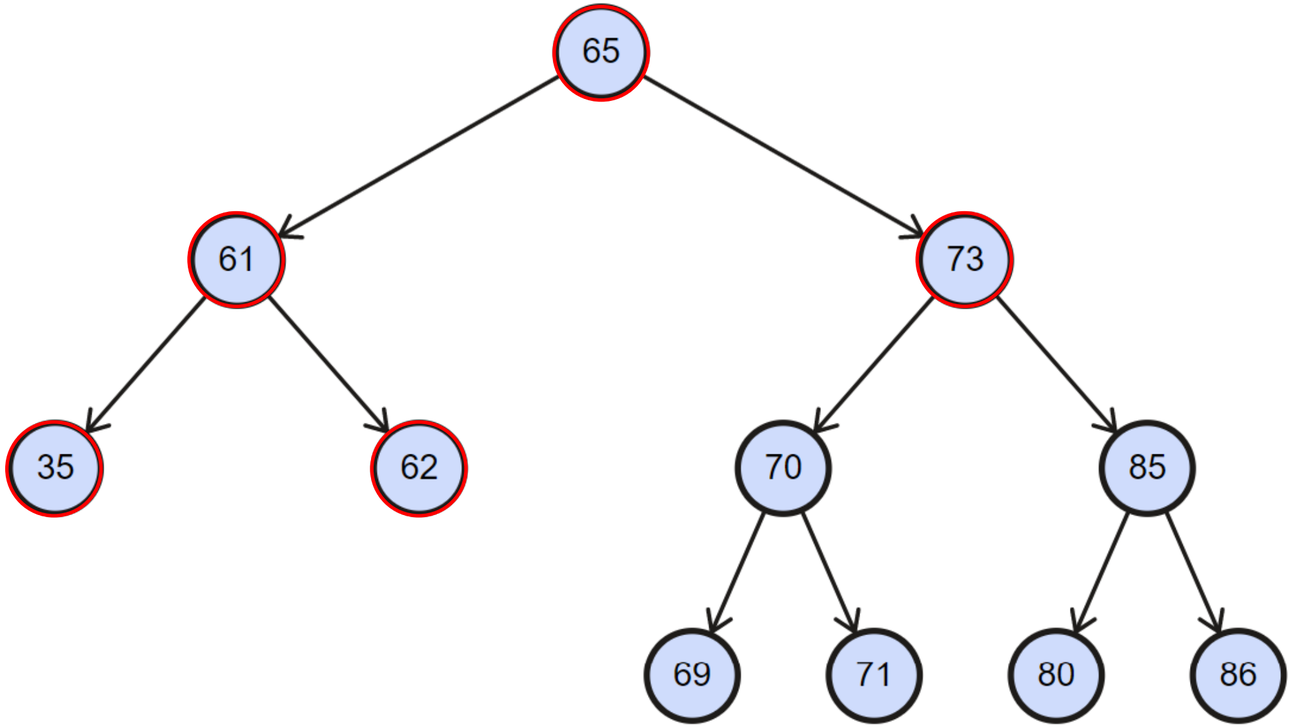
Output: 65, 61, 73, 35, 62

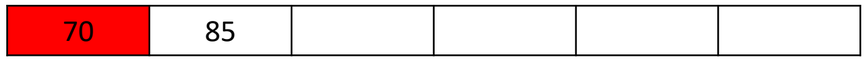
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





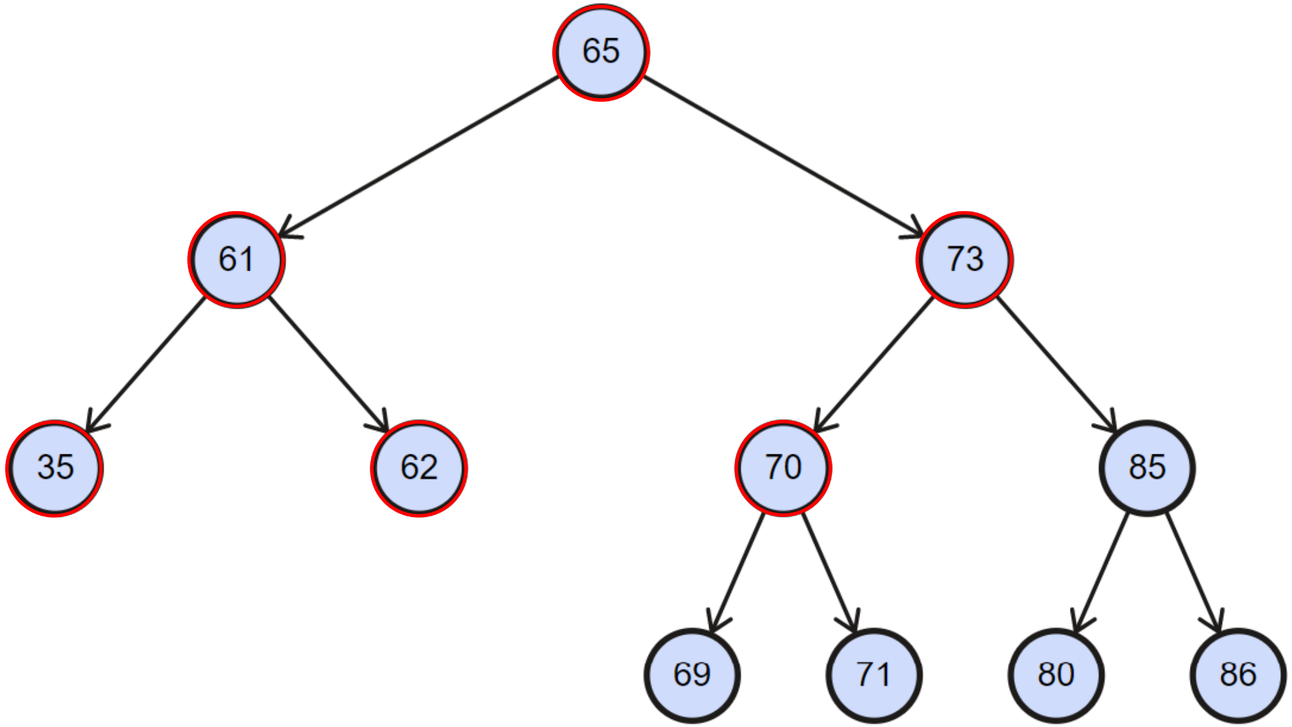
Output: 65, 61, 73, 35, 62

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



85					

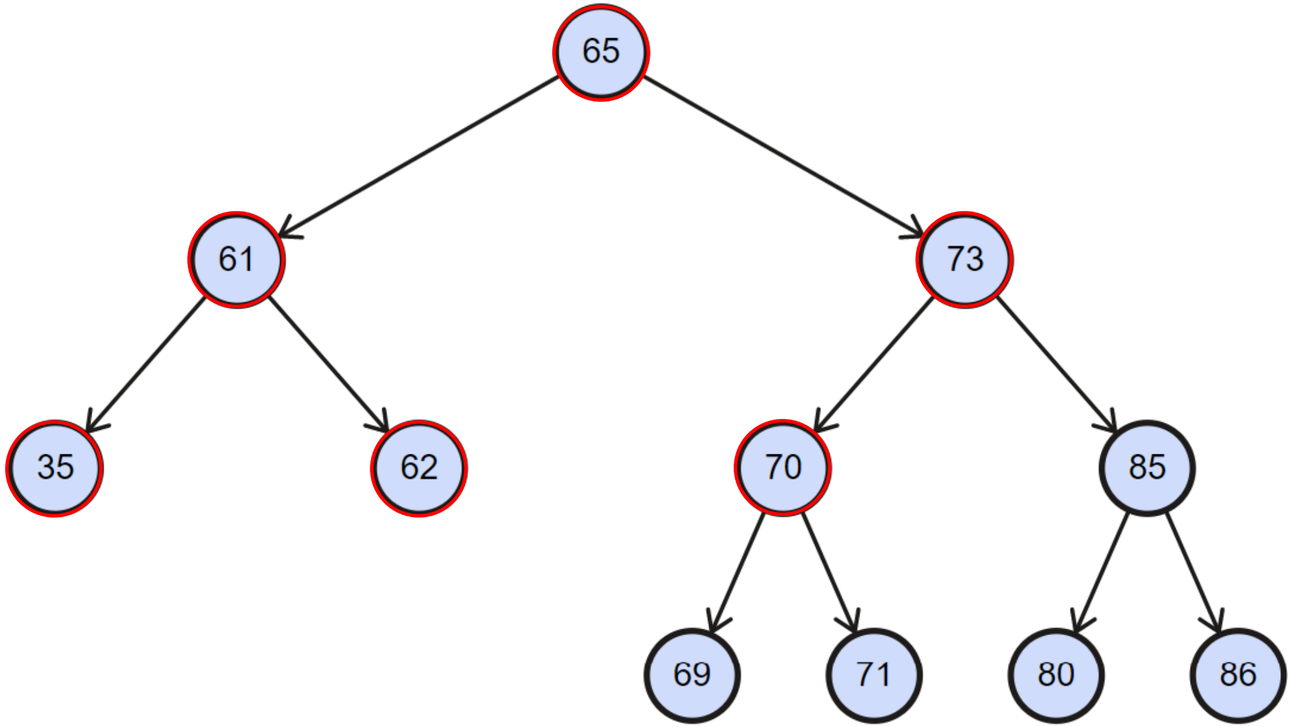
Output: 65, 61, 73, 35, 62, 70

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



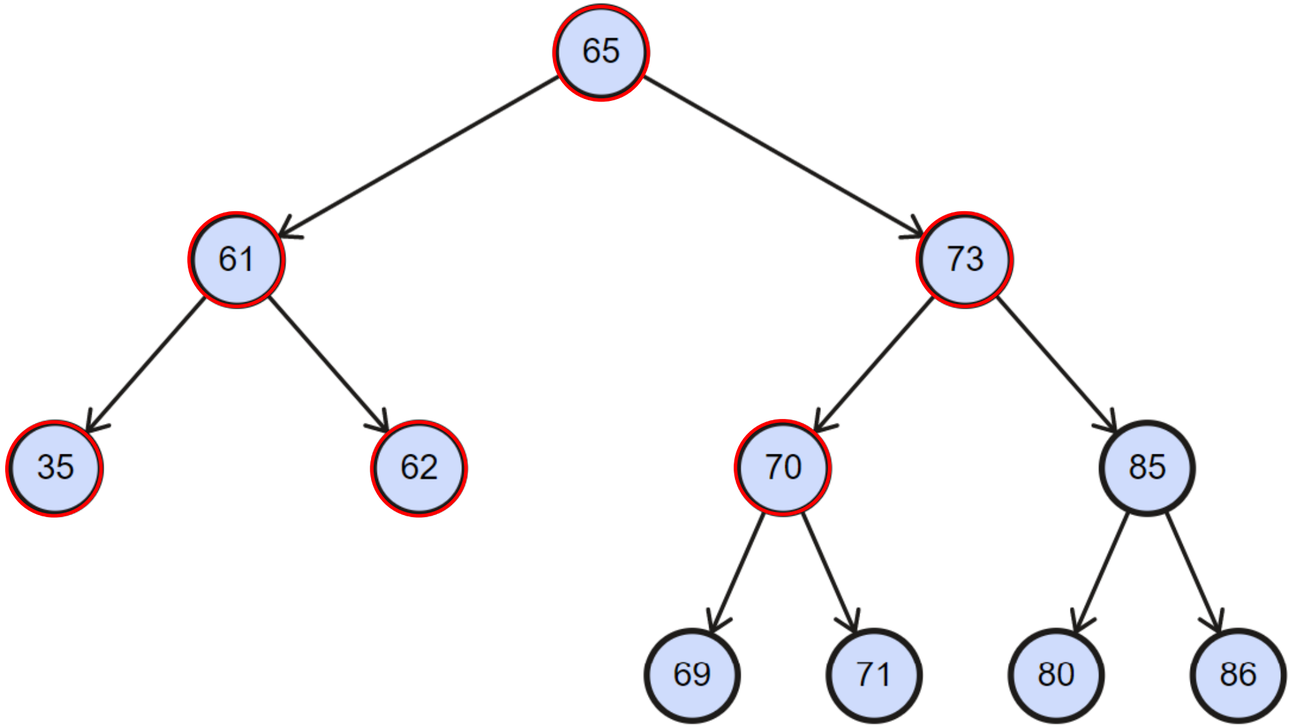
Output: 65, 61, 73, 35, 62, 70

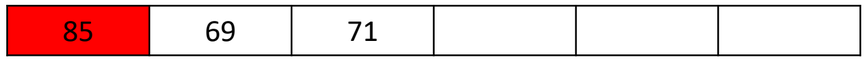
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





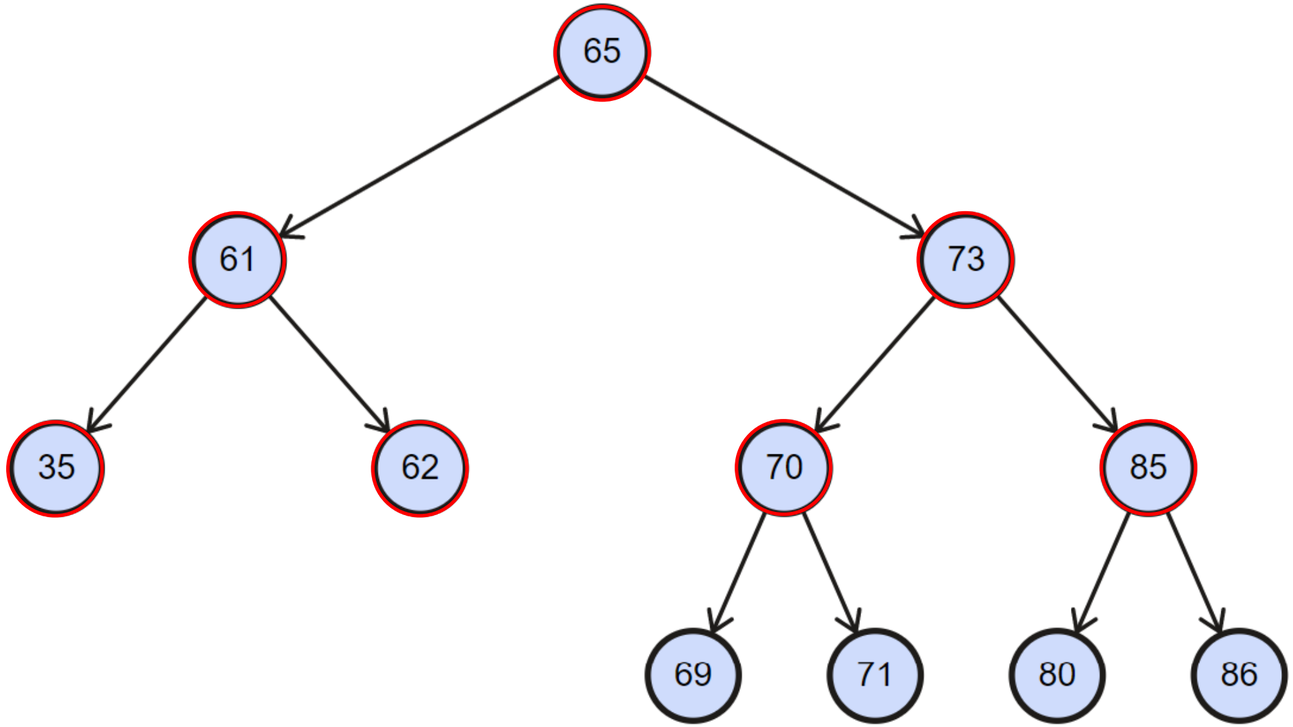
Output: 65, 61, 73, 35, 62, 70

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



69	71				

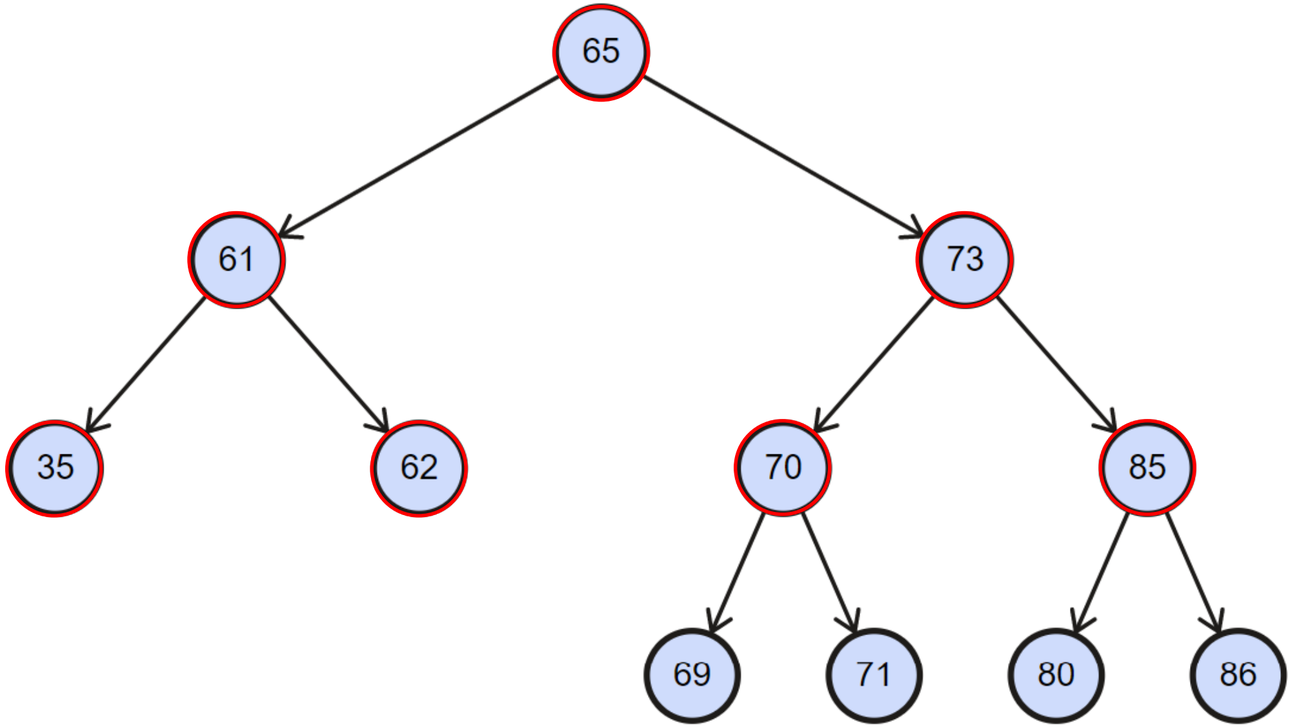
Output: 65, 61, 73, 35, 62, 70, 85

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



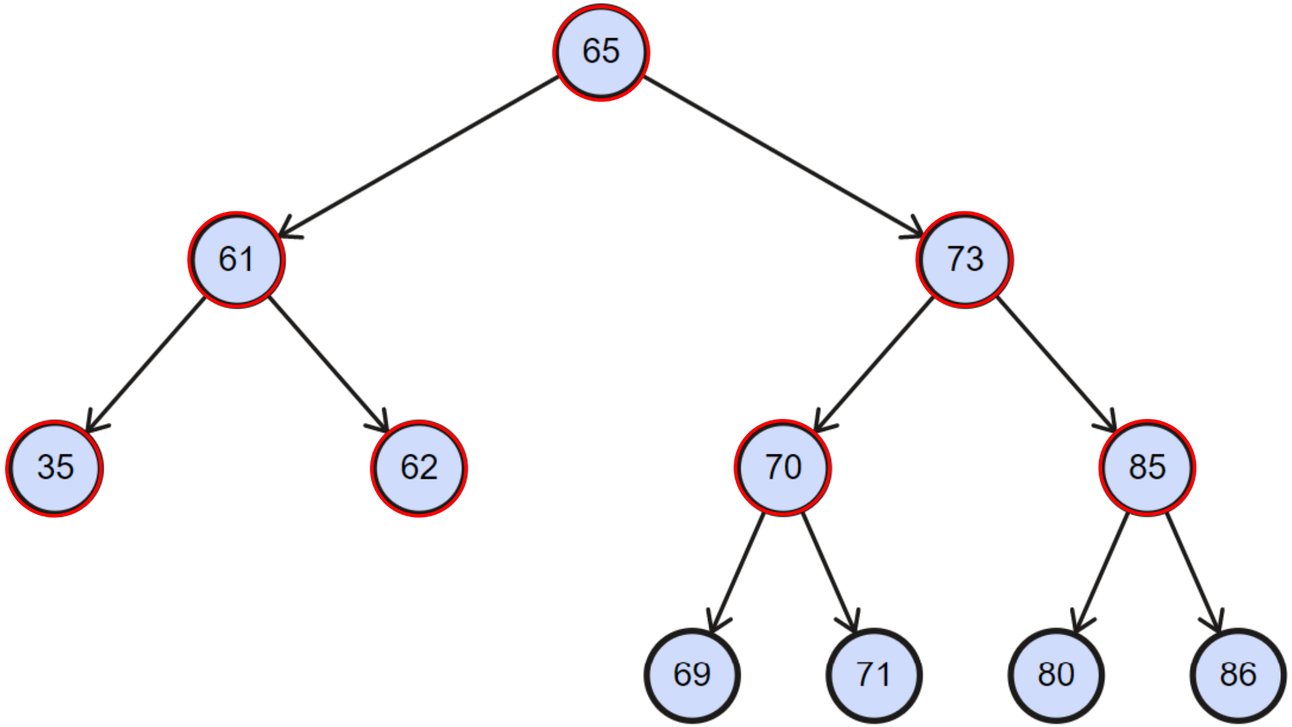
Output: 65, 61, 73, 35, 62, 70, 85

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



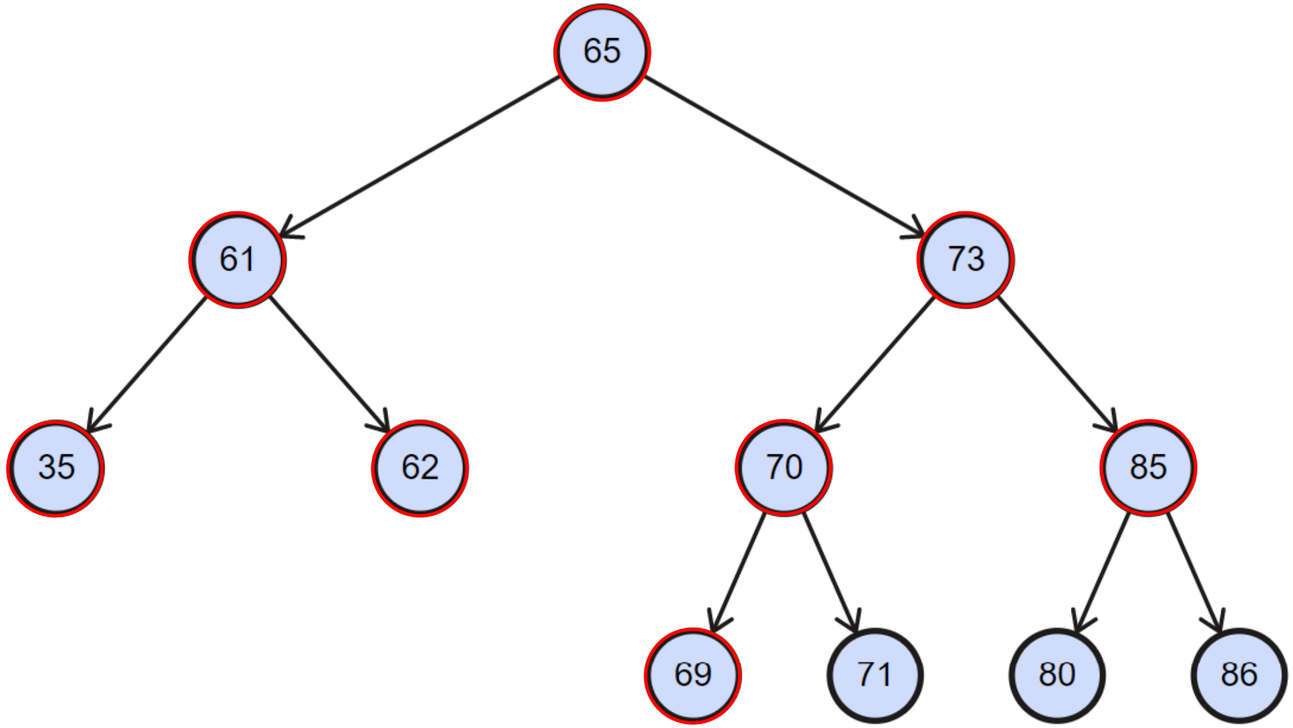
Output: 65, 61, 73, 35, 62, 70, 85

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



71	80	86			

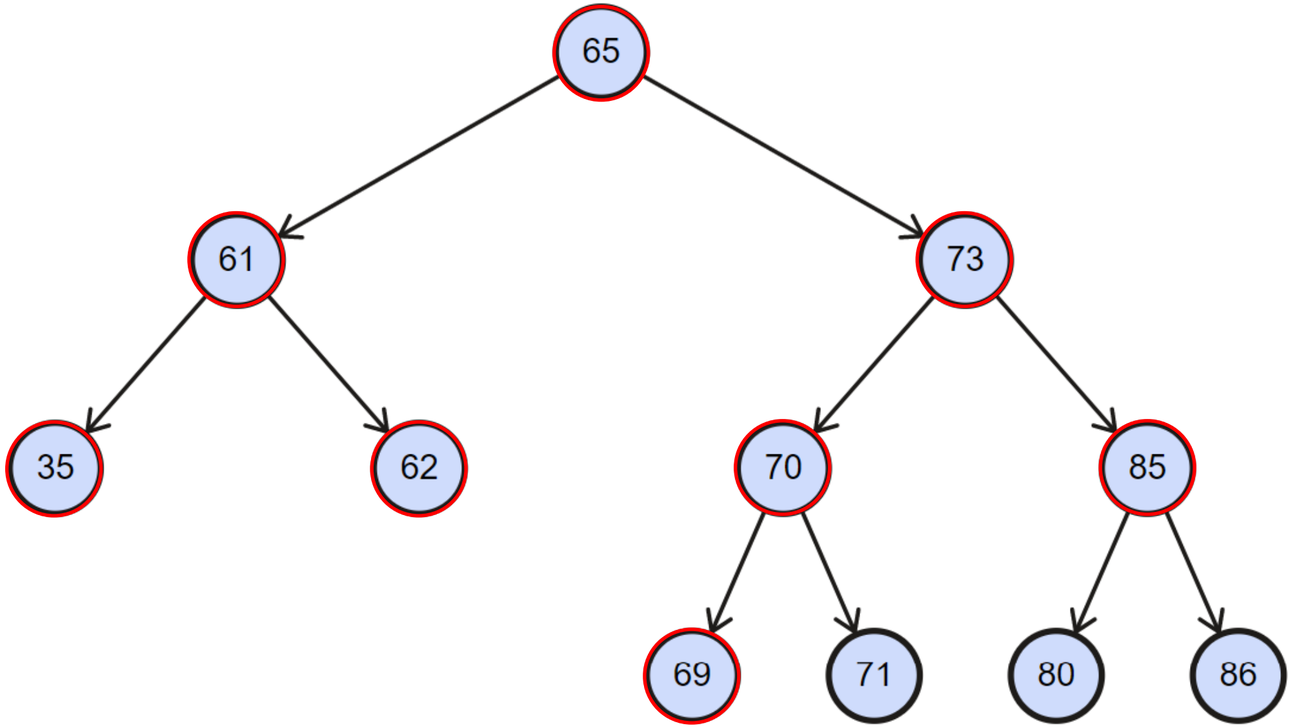
Output: 65, 61, 73, 35, 62, 70, 85, 69

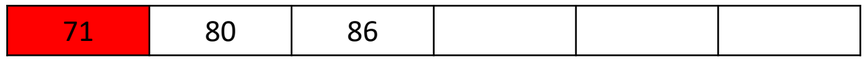
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





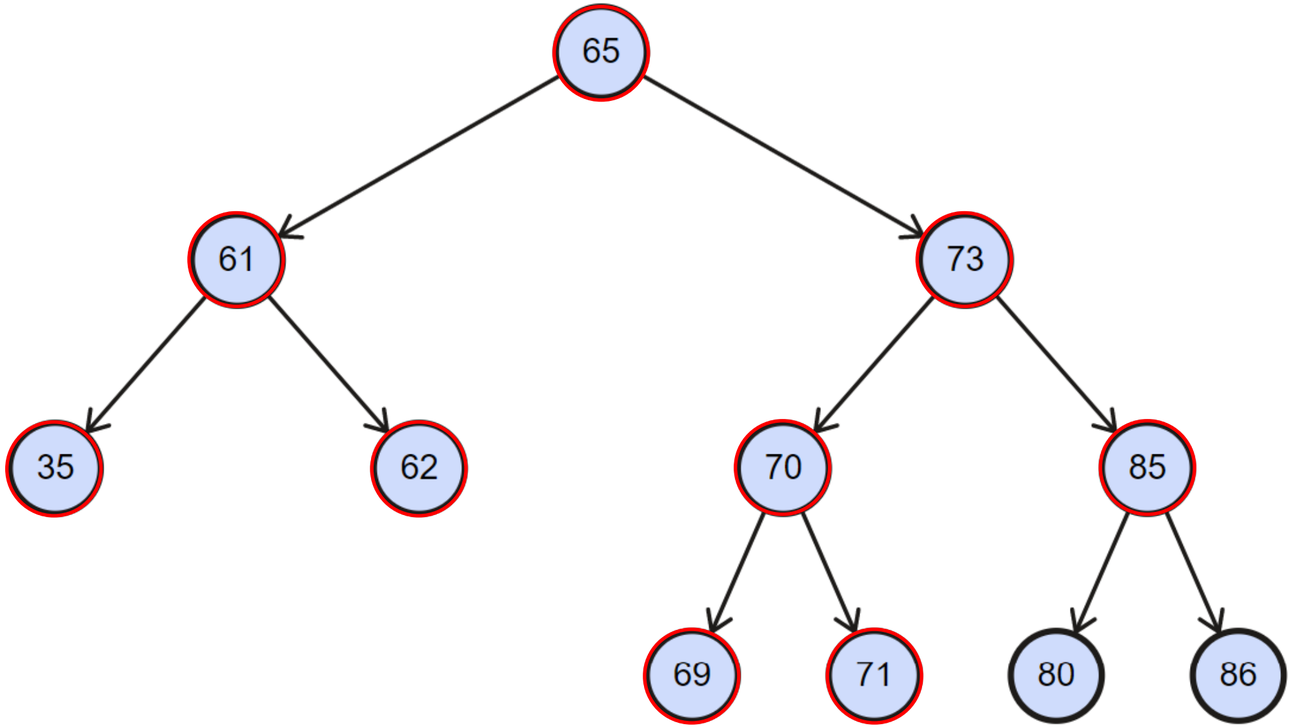
Output: 65, 61, 73, 35, 62, 70, 85, 69

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



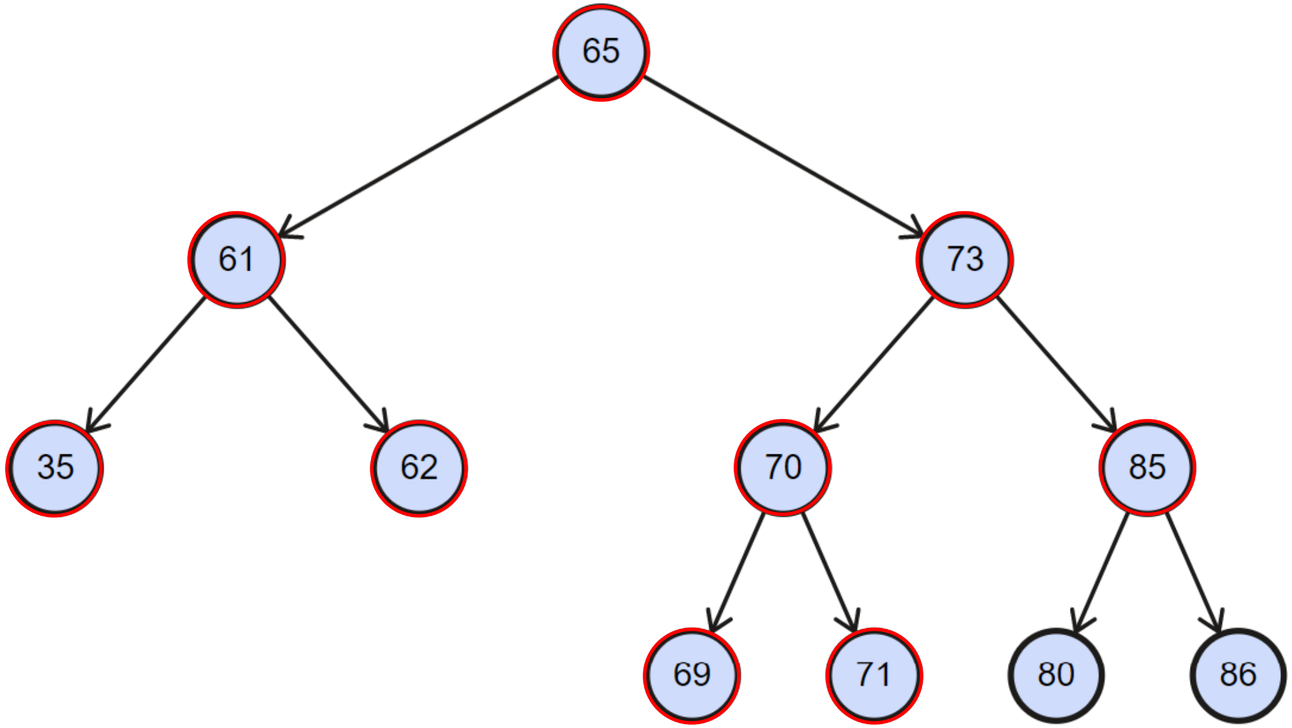
Output: 65, 61, 73, 35, 62, 70, 85, 69, 71

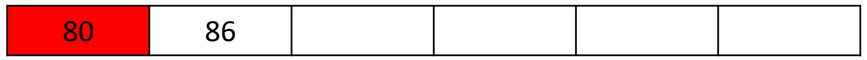
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





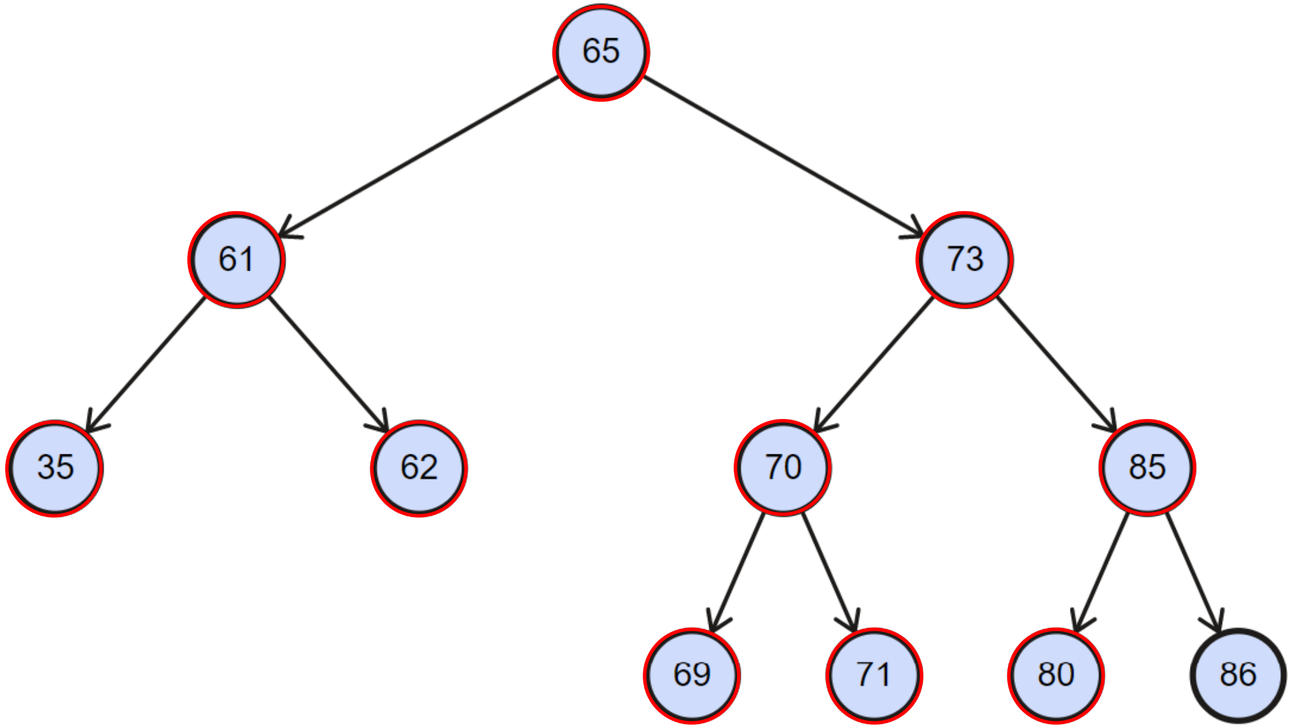
Output: 65, 61, 73, 35, 62, 70, 85, 69, 71

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)



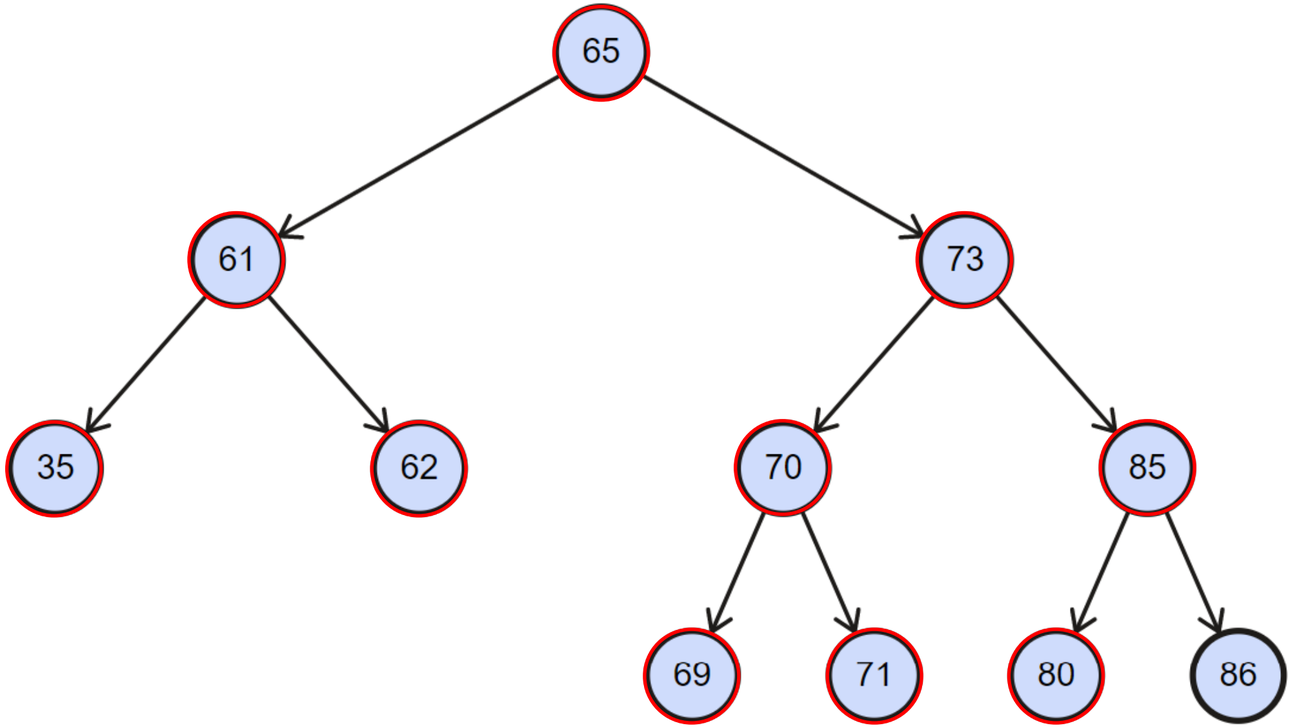
Output: 65, 61, 73, 35, 62, 70, 85, 69, 71, 80

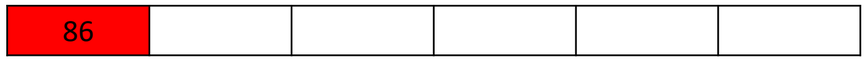
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





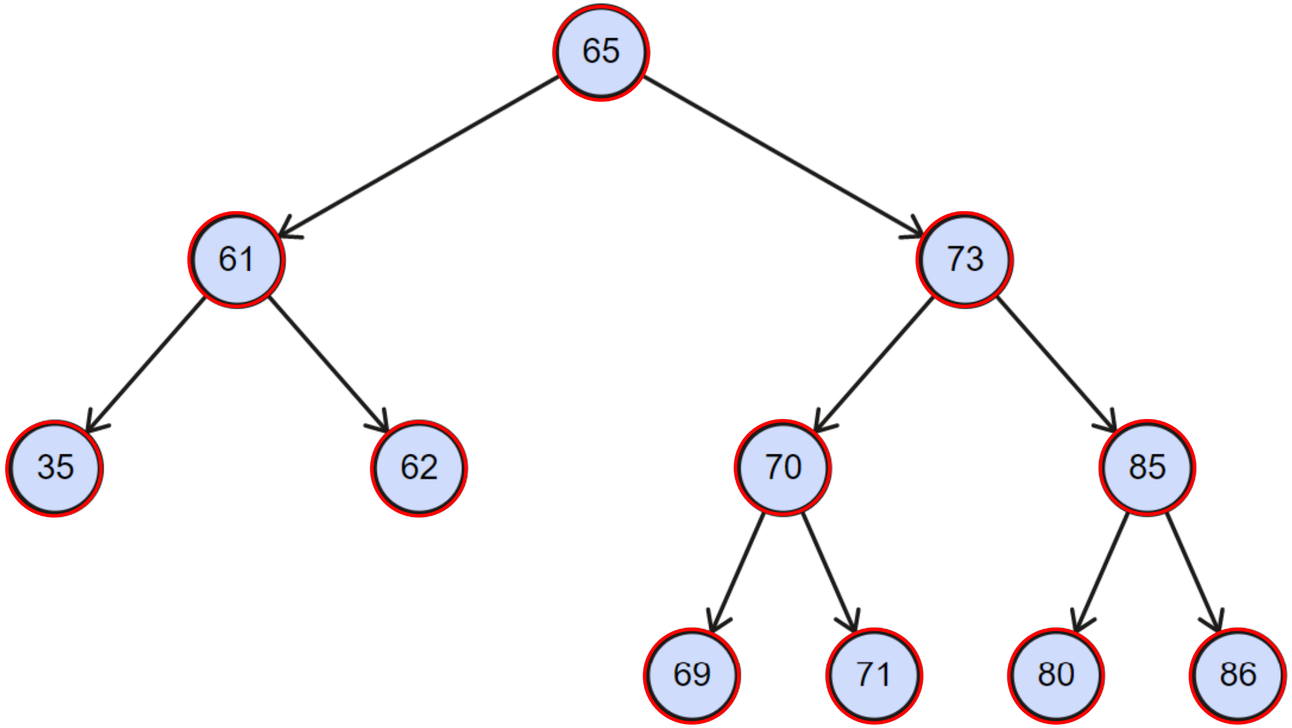
Output: 65, 61, 73, 35, 62, 70, 85, 69, 71, 80

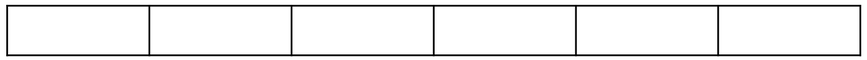
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

1. Parcurgere breadth-first search

- se utilizează o coadă auxiliară
- primul element este adăugat în coadă (push)
- se extrage un element din coadă (pop)
- se procesează elementul extras
- se adaugă fiii săi (stânga, dreapta) în coadă (push)





Output: 65, 61, 73, 35, 62, 70, 85, 69, 71, 80, 86

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

- **preordine:** se vizitează mai întâi rădăcina, copilul (copiii) din stânga, iar apoi copilul (copiii) din dreapta;
- **inordine:** se vizitează mai întâi copilul (copiii) din stânga, apoi rădăcina și copilul (copiii) din dreapta;
- **postordine:** se vizitează copilul (copiii) din stânga, apoi copilul (copiii) din dreapta și apoi rădăcina.

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

- **preordine:** se vizitează mai întâi rădăcina, copilul (copiii) din stânga, iar apoi copilul (copiii) din dreapta;
- **inordine:** se vizitează mai întâi copilul (copiii) din stânga, apoi rădăcina și copilul (copiii) din dreapta;
- **postordine:** se vizitează copilul (copiii) din stânga, apoi copilul (copiii) din dreapta și apoi rădăcina.

```
/* parcurgere arbore in preordine */
void afisare_preordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere radacina, stanga, dreapta */
  printf("%d \n", prim->x);
  afisare_preordine(prim->NOD_stanga);
  afisare_preordine(prim->NOD_dreapta);
 }
}
```

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

- **preordine:** se vizitează mai întâi rădăcina, copilul (copiii) din stânga, iar apoi copilul (copiii) din dreapta;
- **inordine:** se vizitează mai întâi copilul (copiii) din stânga, apoi rădăcina și copilul (copiii) din dreapta;
- **postordine:** se vizitează copilul (copiii) din stânga, apoi copilul (copiii) din dreapta și apoi rădăcina.

```
/* parcurgere arbore in preordine */
void afisare_preordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere radacina, stanga, dreapta */
  printf("%d \n", prim->x);
  afisare_preordine(prim->NOD_stanga);
  afisare_preordine(prim->NOD_dreapta);
 }
}
```

```
/* parcurgere arbore in inordine */
void afisare_inordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere stanga, radacina, dreapta */
  afisare_inordine(prim->NOD_stanga);
  printf("%d \n", prim->x);
  afisare_inordine(prim->NOD_dreapta);
 }
}
```

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

- **preordine:** se vizitează mai întâi rădăcina, copilul (copiii) din stânga, iar apoi copilul (copiii) din dreapta;
- **inordine:** se vizitează mai întâi copilul (copiii) din stânga, apoi rădăcina și copilul (copiii) din dreapta;
- **postordine:** se vizitează copilul (copiii) din stânga, apoi copilul (copiii) din dreapta și apoi rădăcina.

```
/* parcurgere arbore in preordine */
void afisare_preordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere radacina, stanga, dreapta */
  printf("%d \n", prim->x);
  afisare_preordine(prim->NOD_stanga);
  afisare_preordine(prim->NOD_dreapta);
 }
}
```

```
/* parcurgere arbore in inordine */
void afisare_inordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere stanga, radacina, dreapta */
  afisare_inordine(prim->NOD_stanga);
  printf("%d \n", prim->x);
  afisare_inordine(prim->NOD_dreapta);
 }
}
```

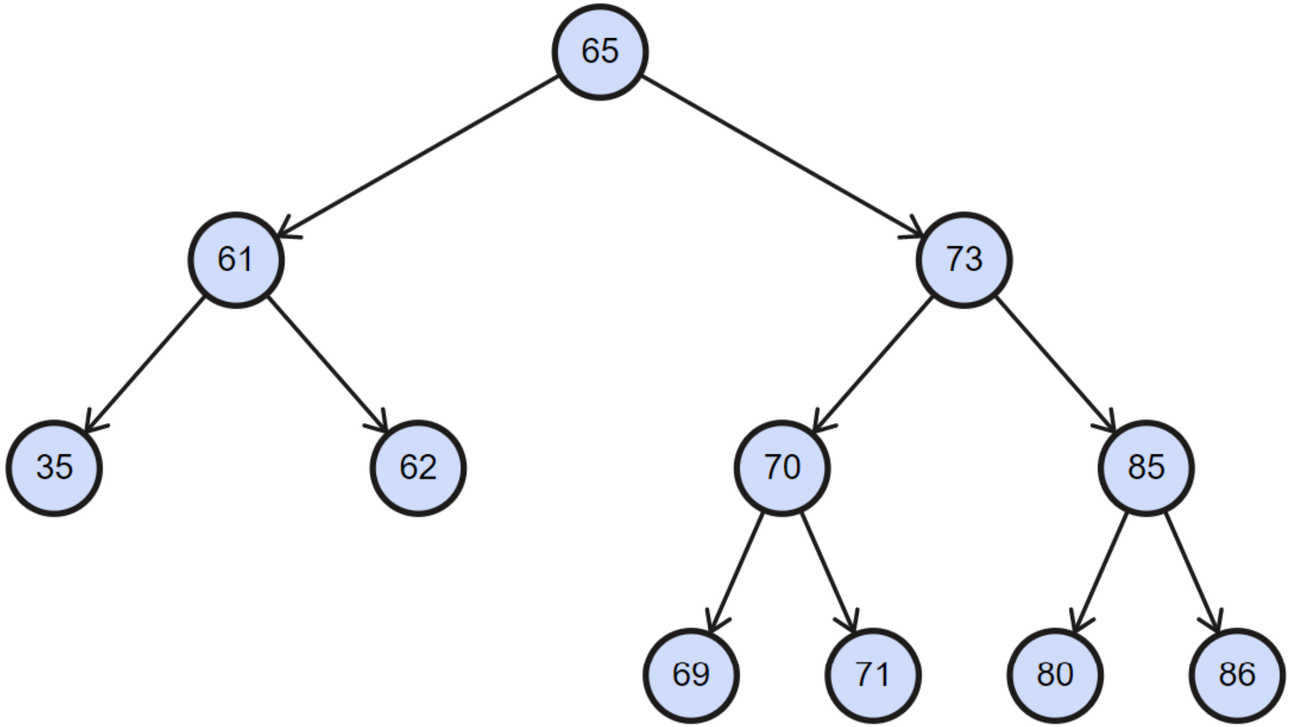
```
/* parcurgere arbore in postordine */
void afisare_postordine(struct NOD *prim)
{
 if (prim!=NULL)
 {
/* parcurgere stanga, dreapta, radacina */
  afisare_postordine(prim->NOD_stanga);
  afisare_postordine(prim->NOD_dreapta);
  printf("%d \n", prim->x);
 }
```

Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine :

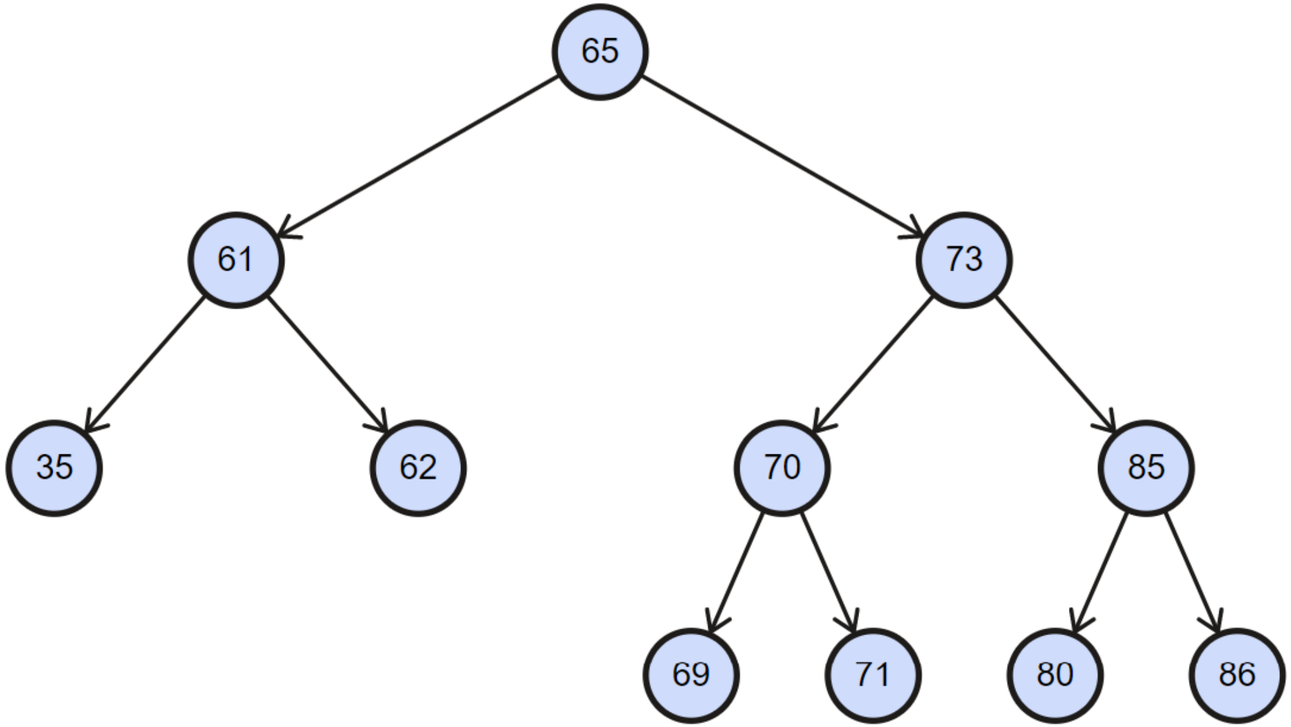


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65,

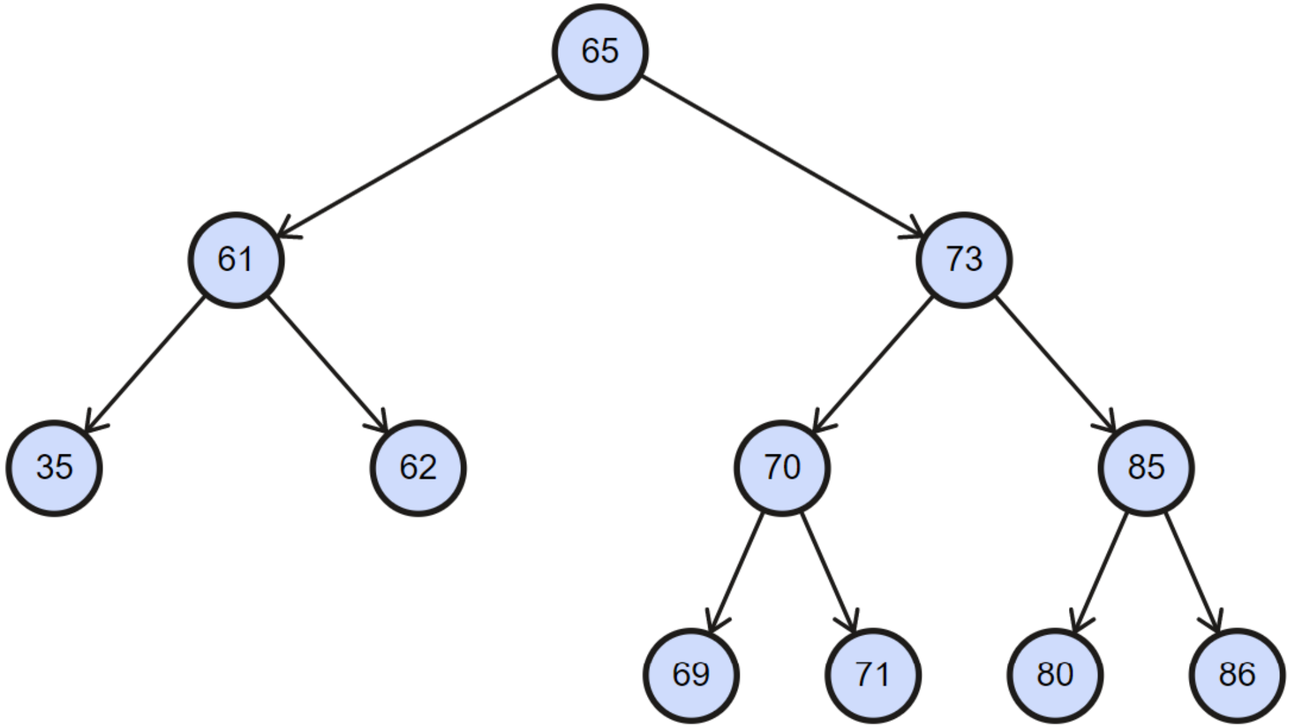


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61,

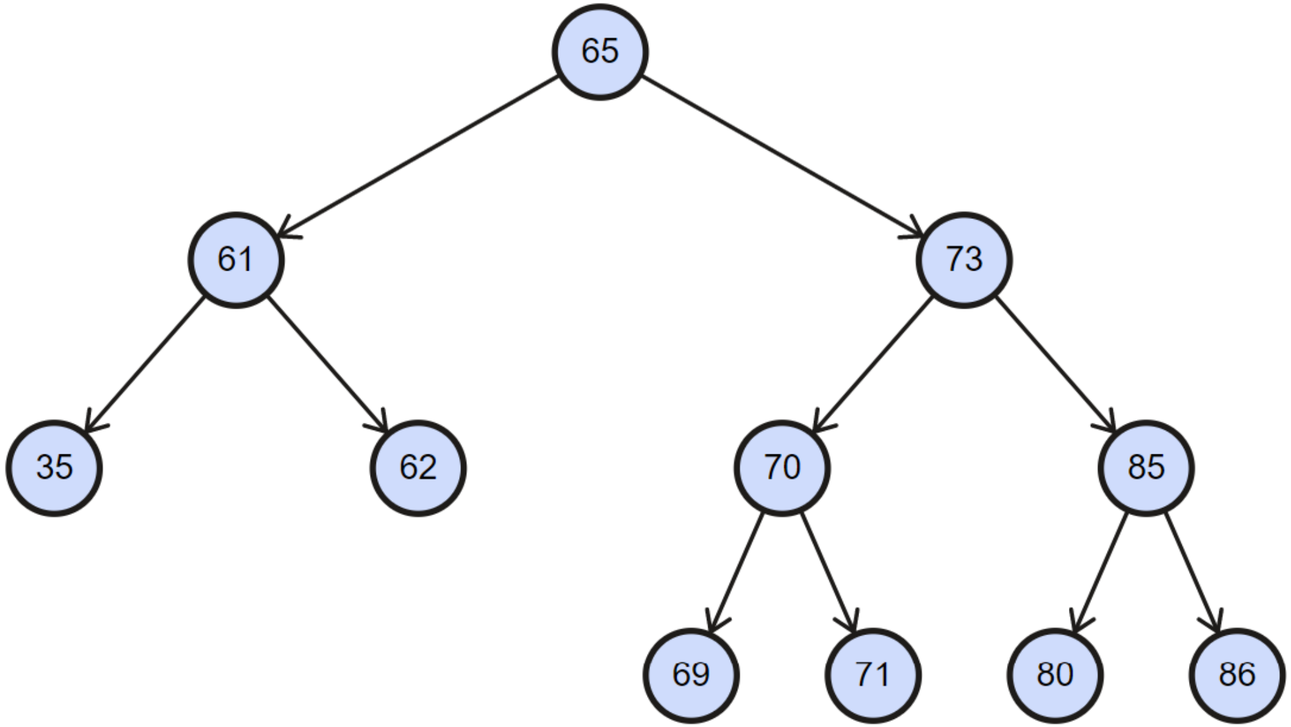


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35,

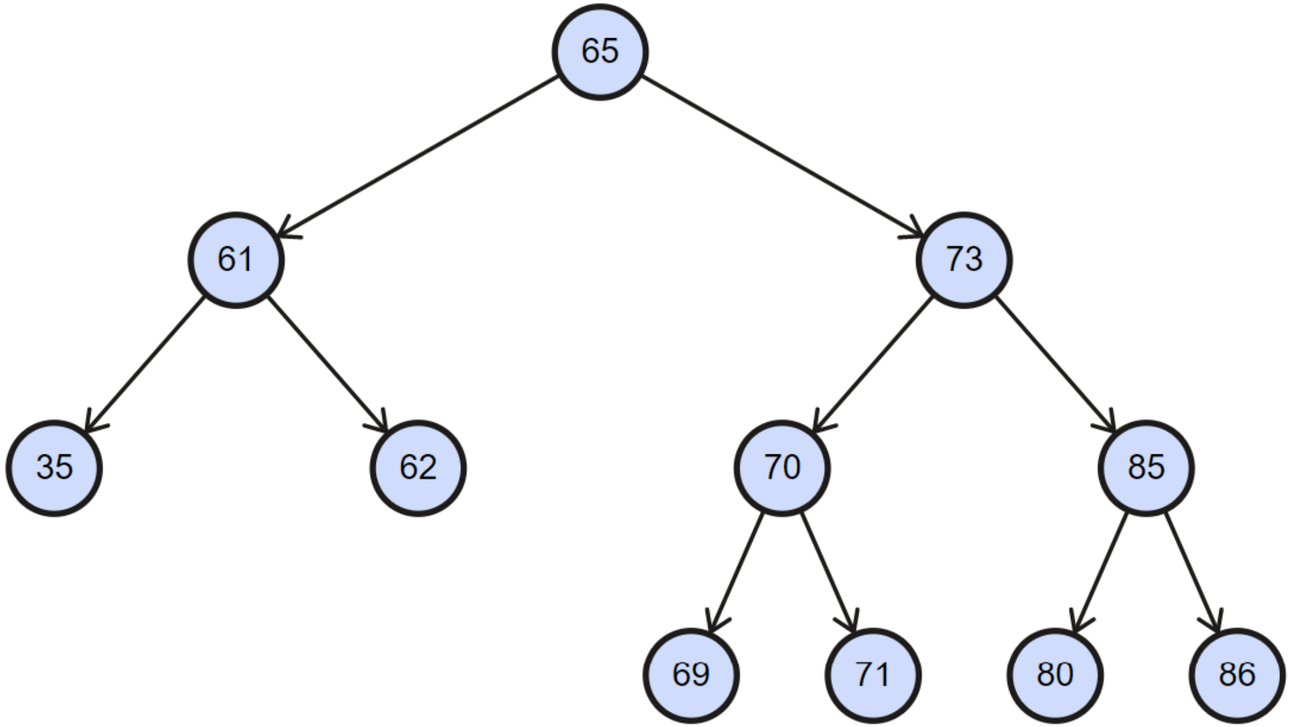


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62,

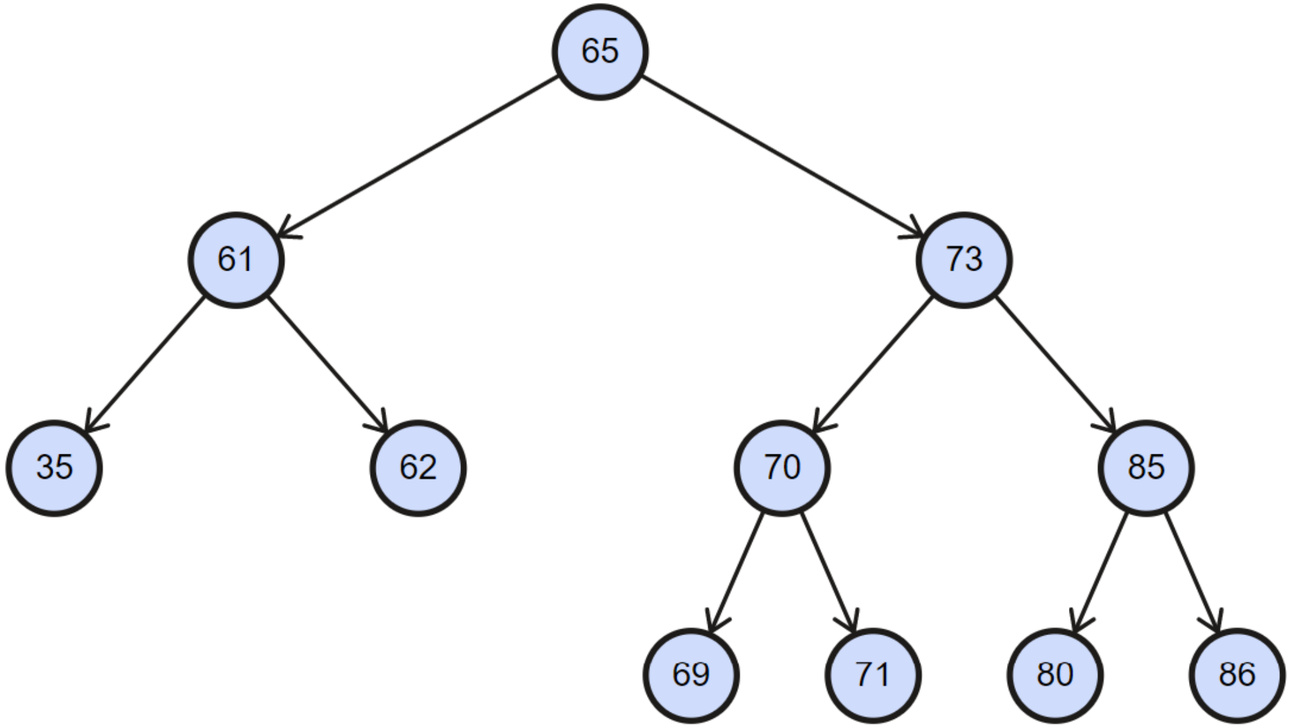


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73,

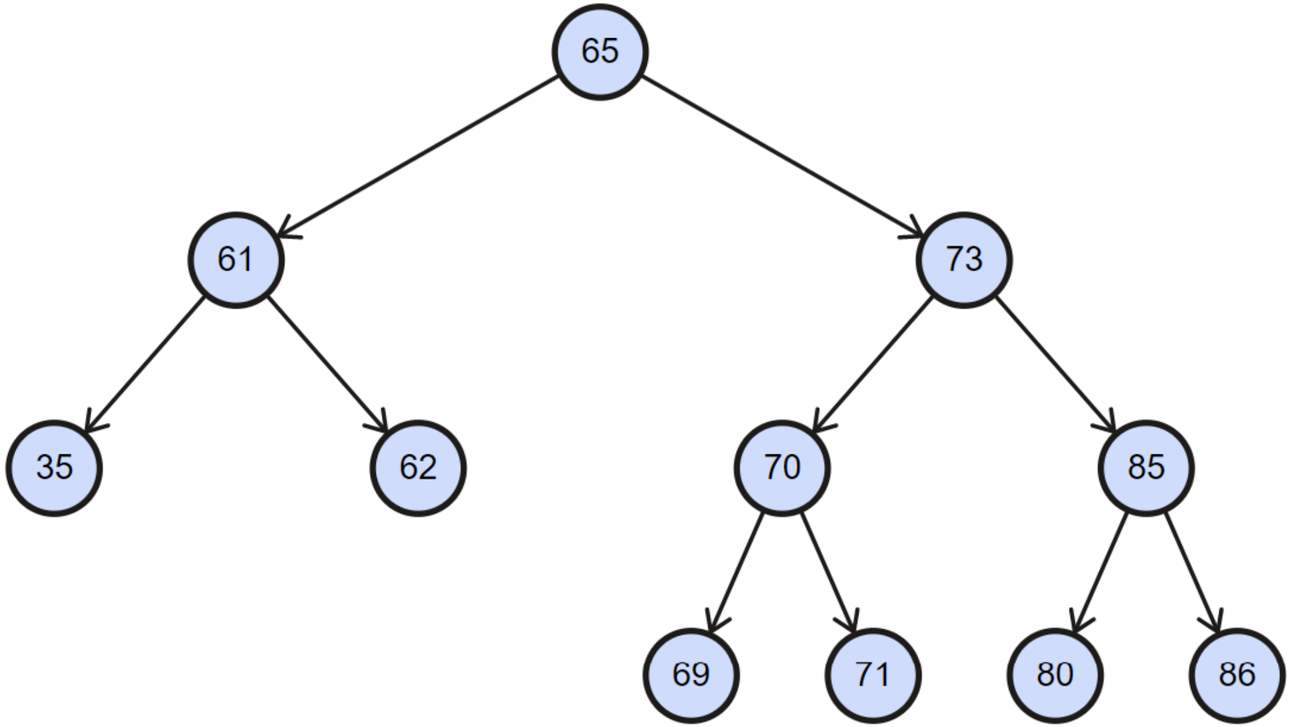


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70,

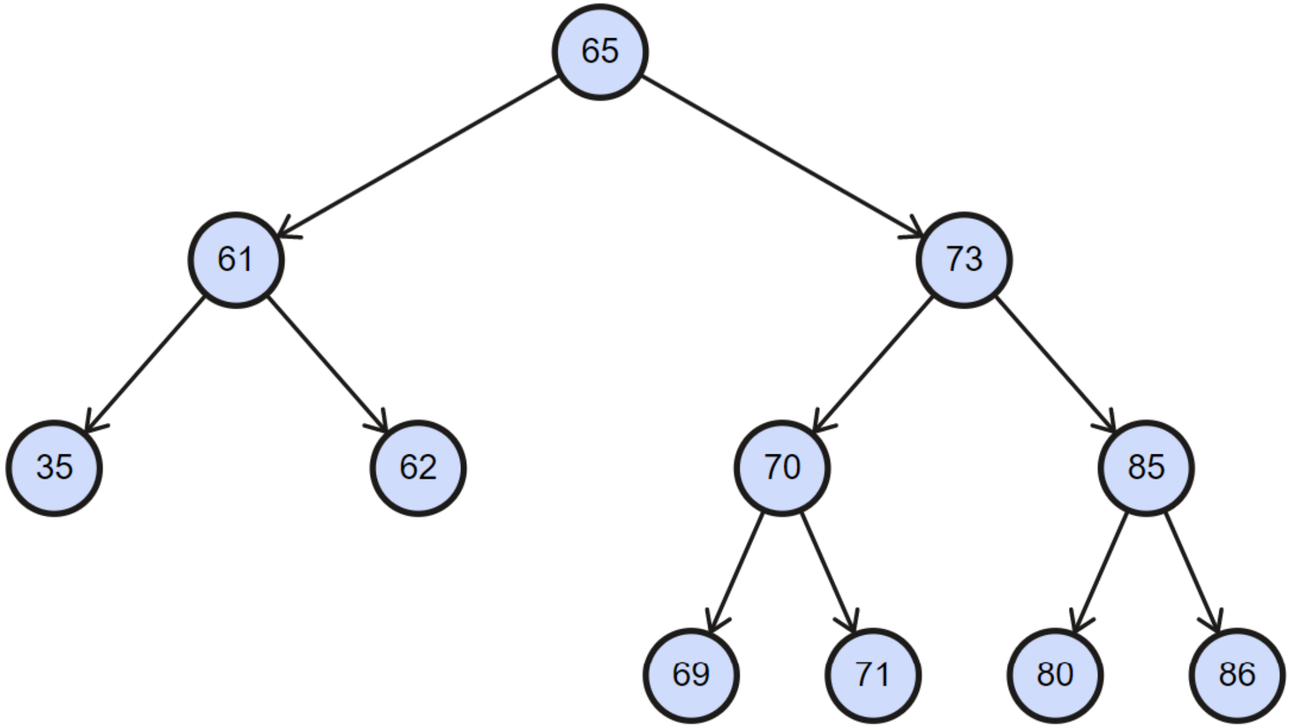


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69,

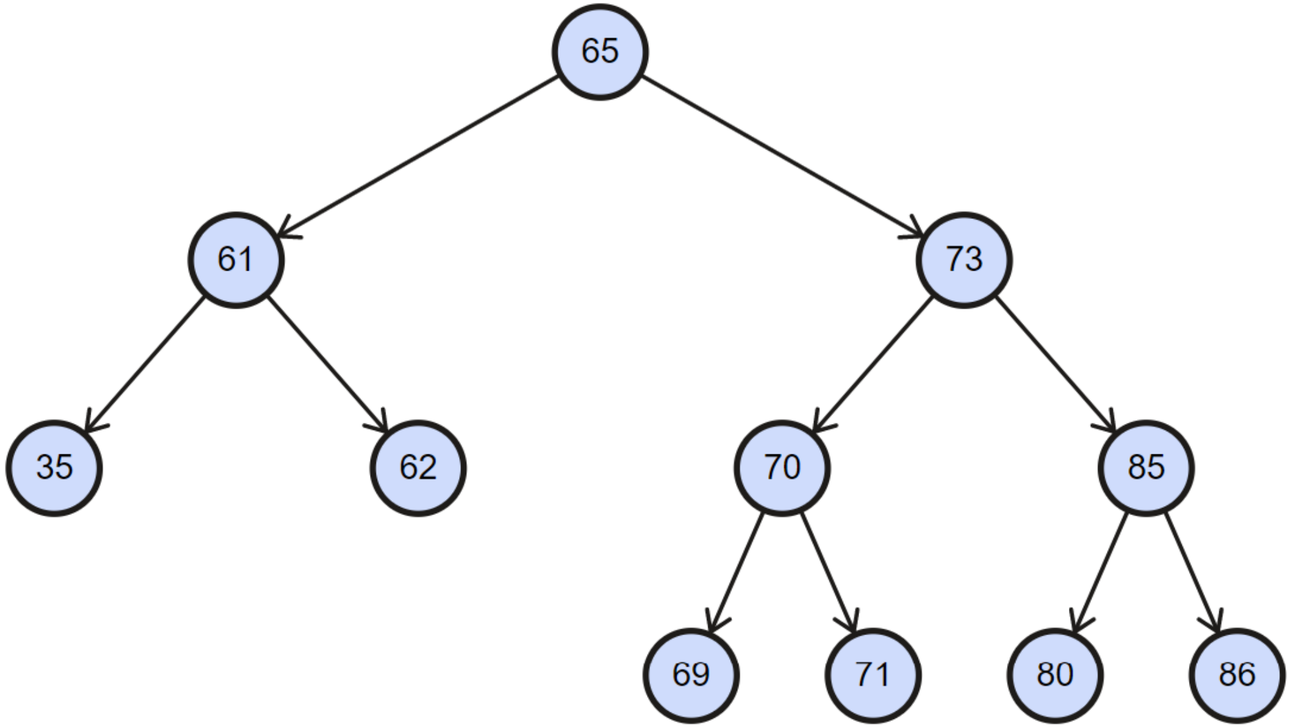


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71,

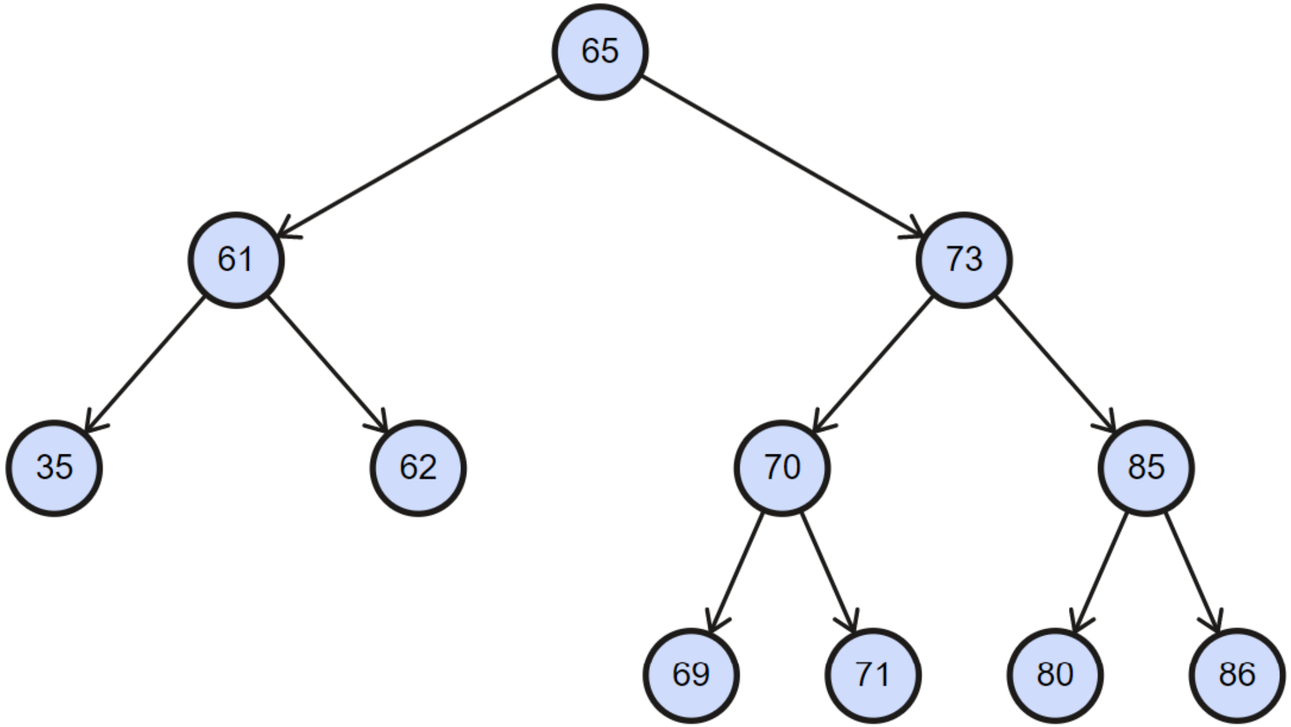


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85,

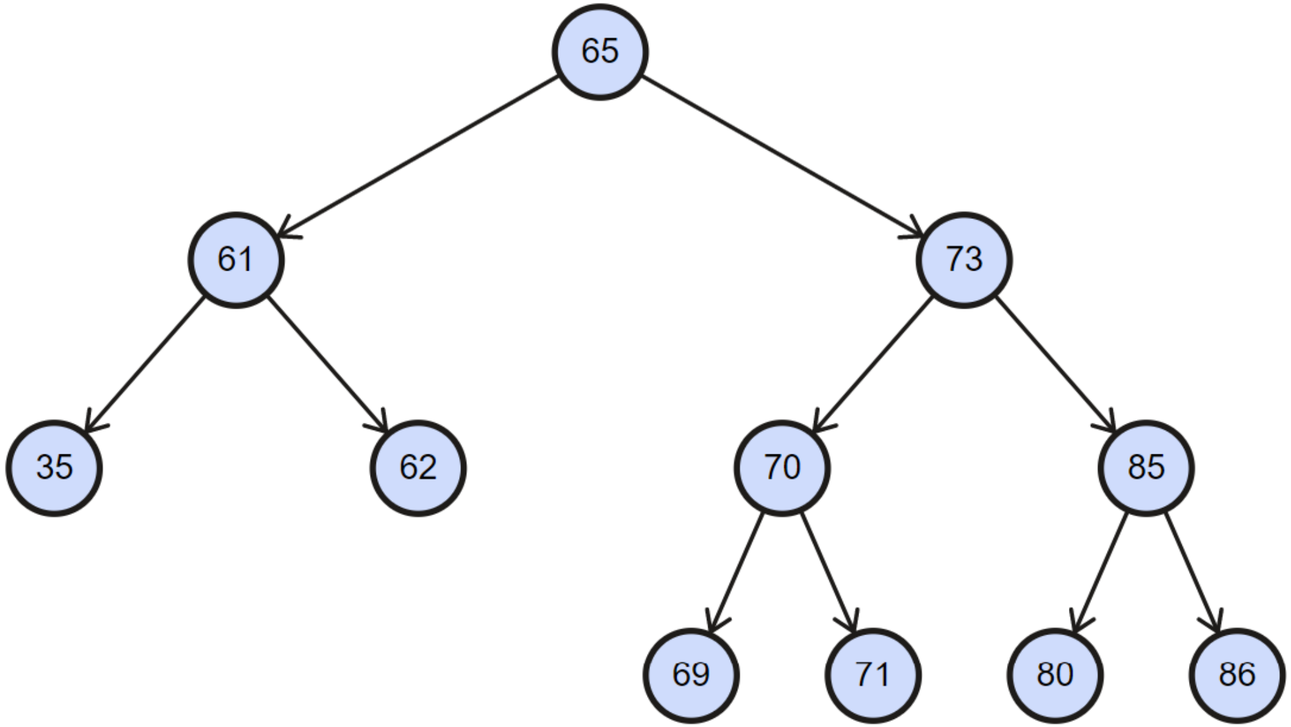


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80,

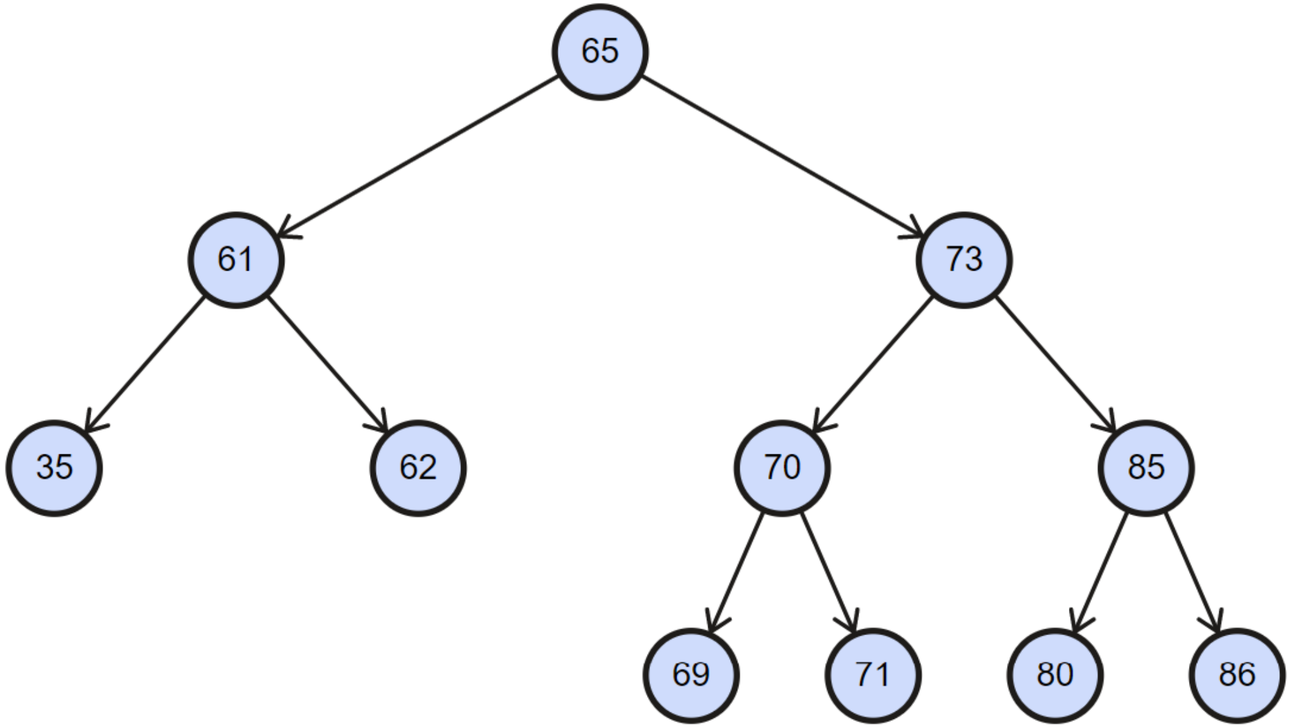


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86



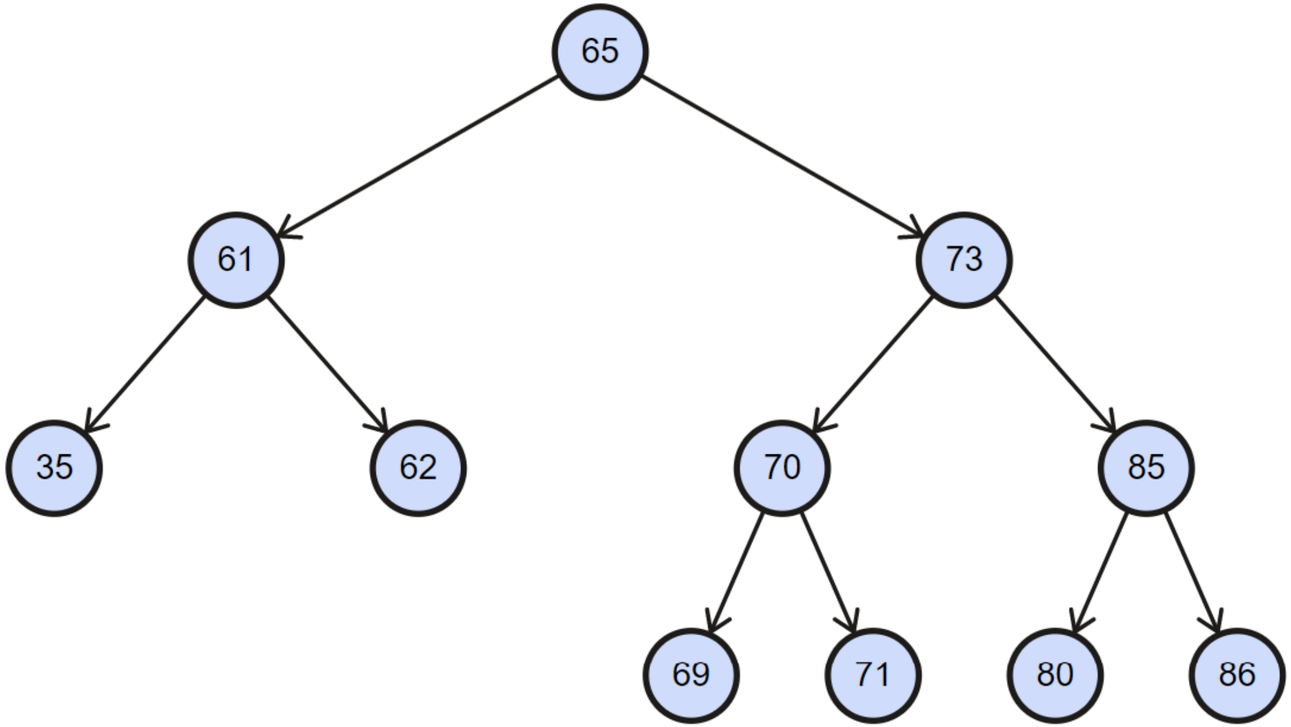
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine :



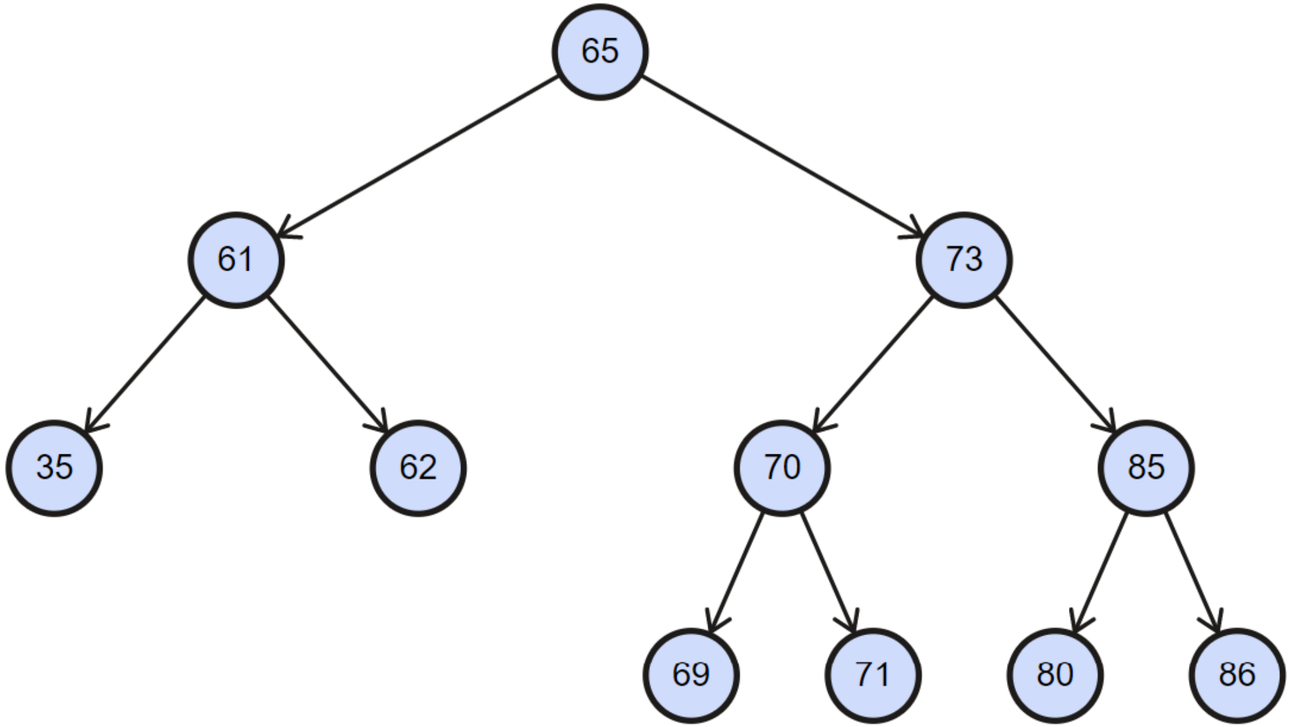
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35,



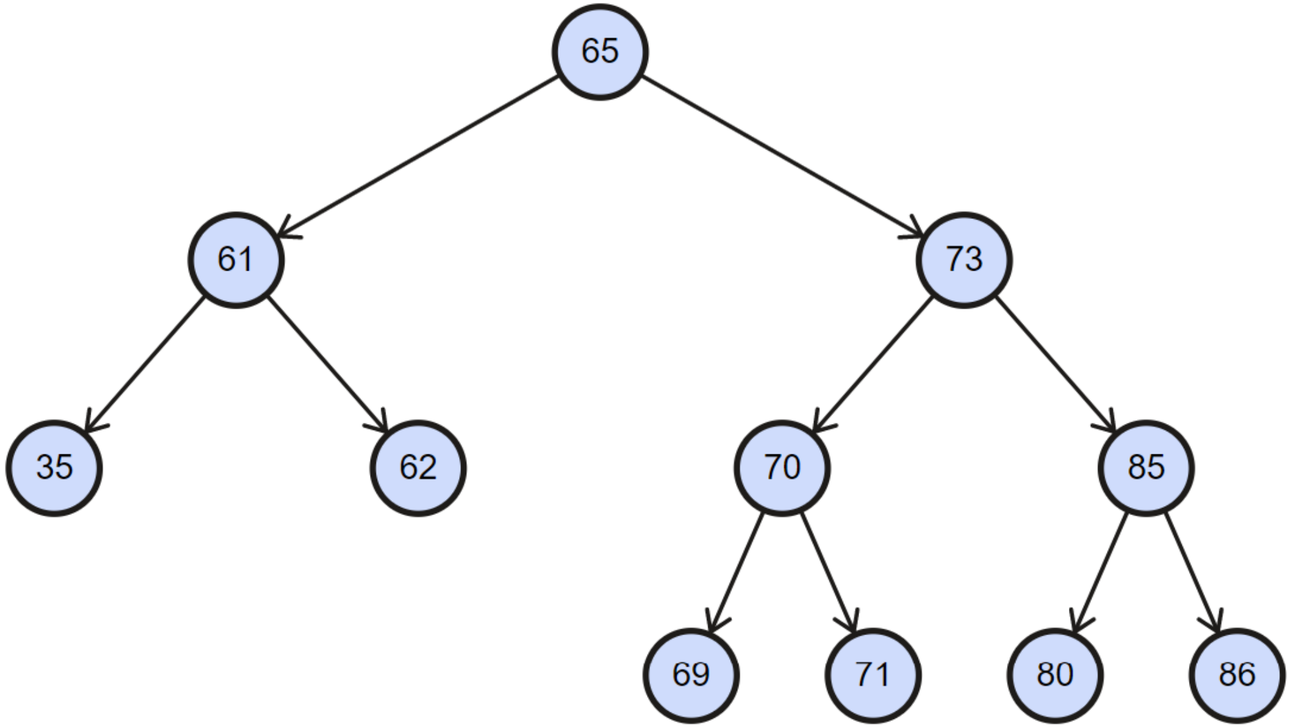
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61,



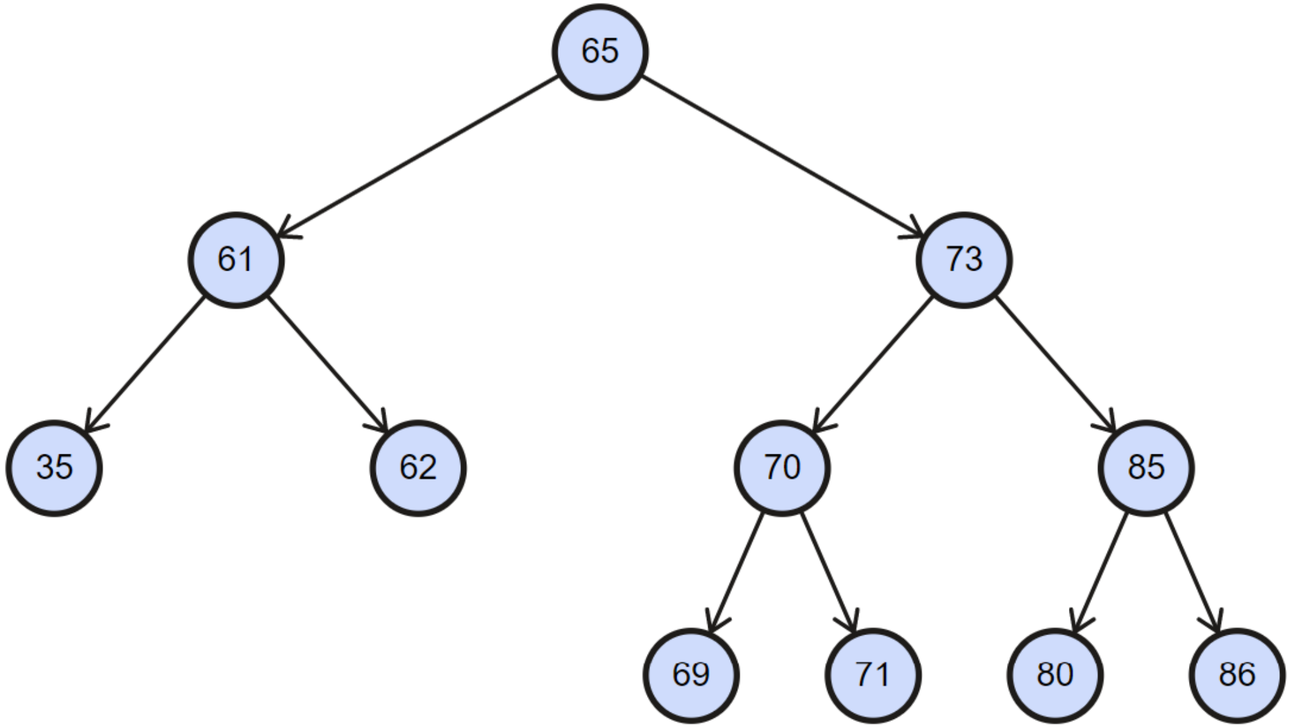
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62,



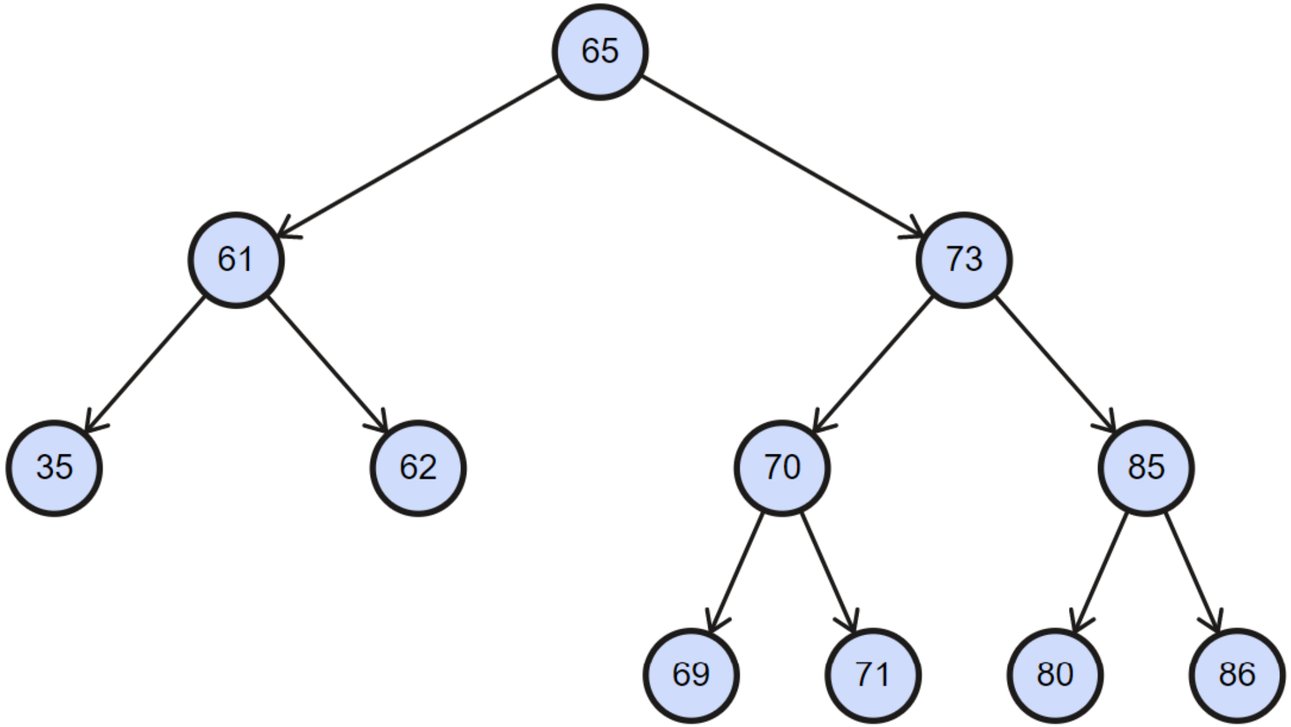
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65,



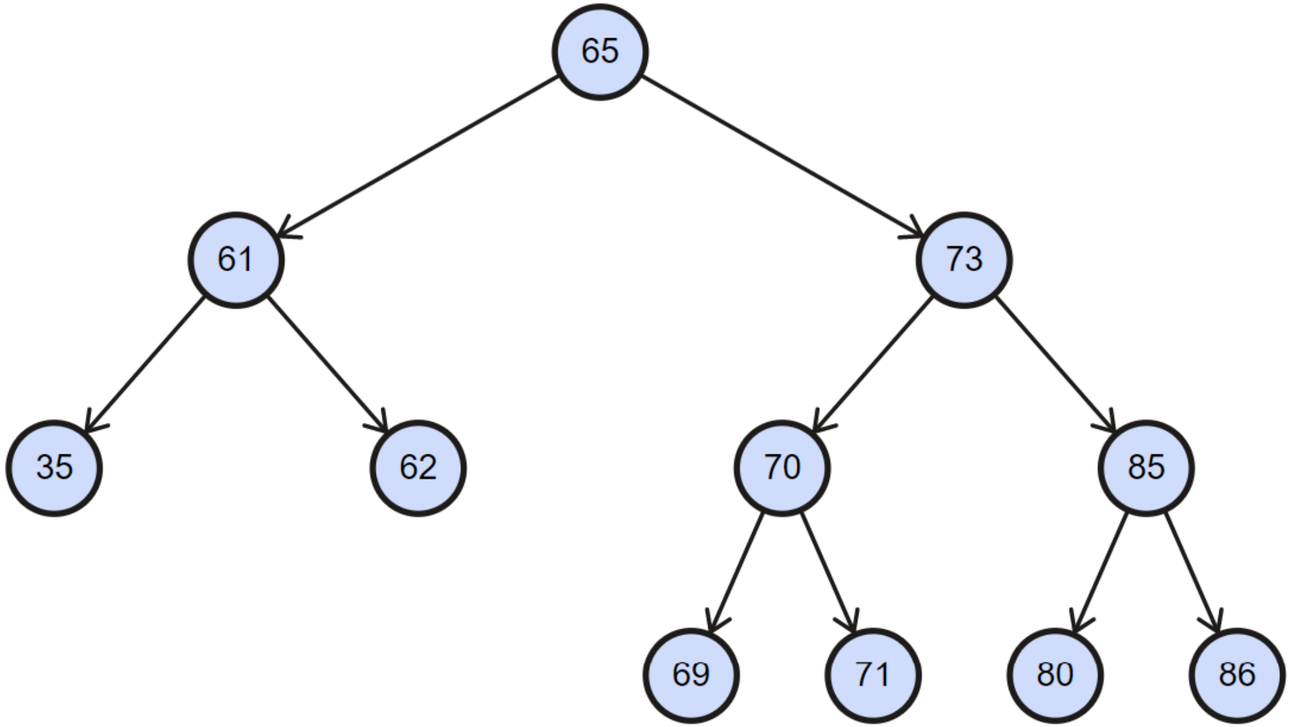
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69,



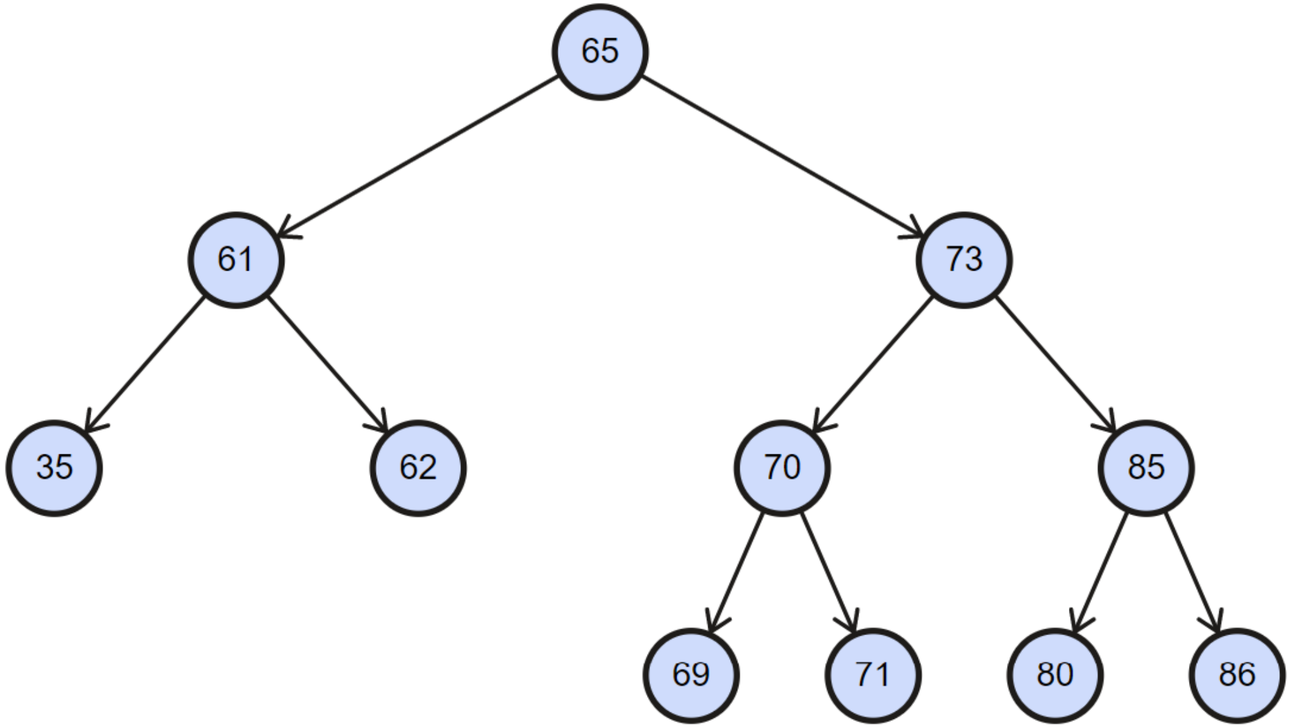
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70,



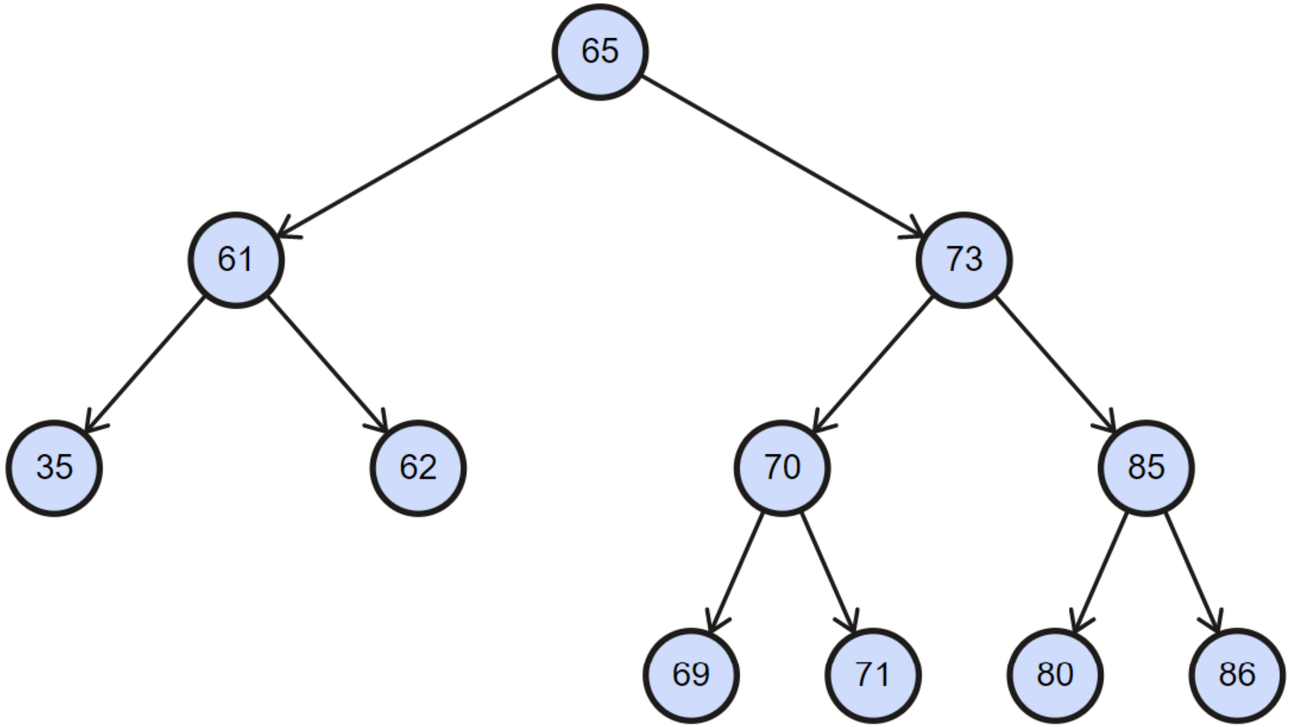
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71,



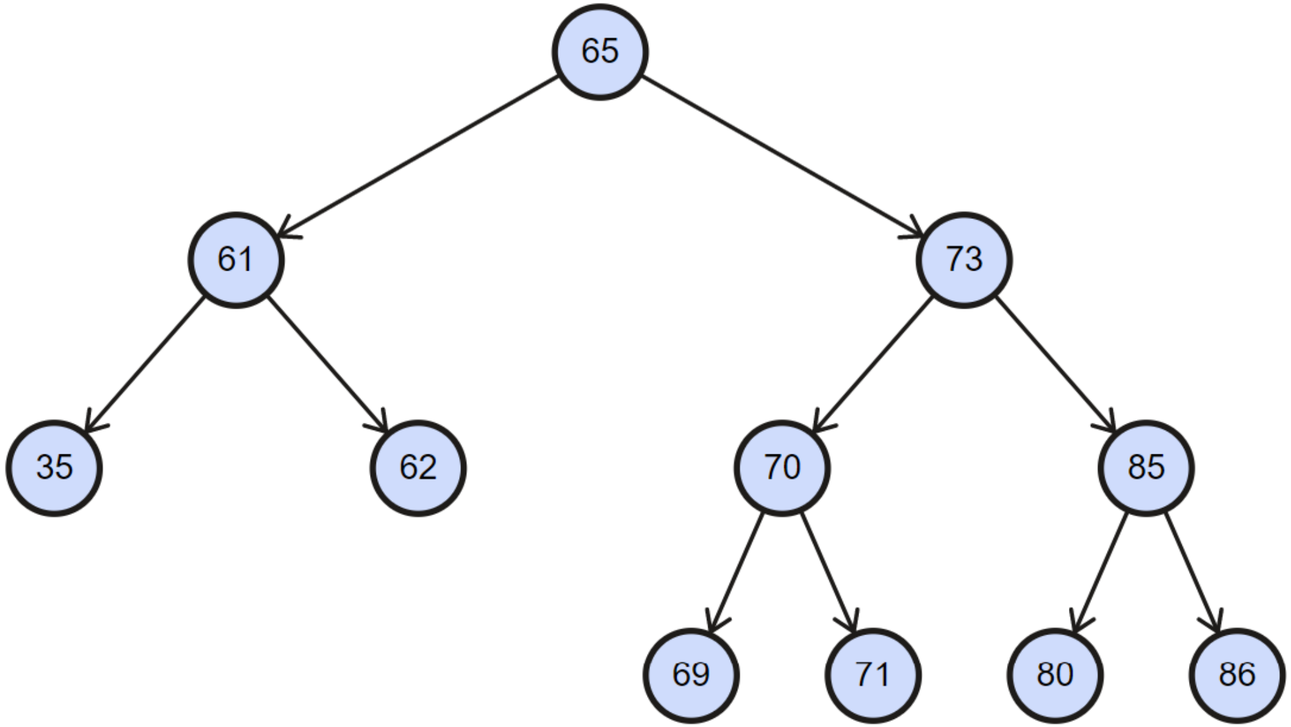
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73,



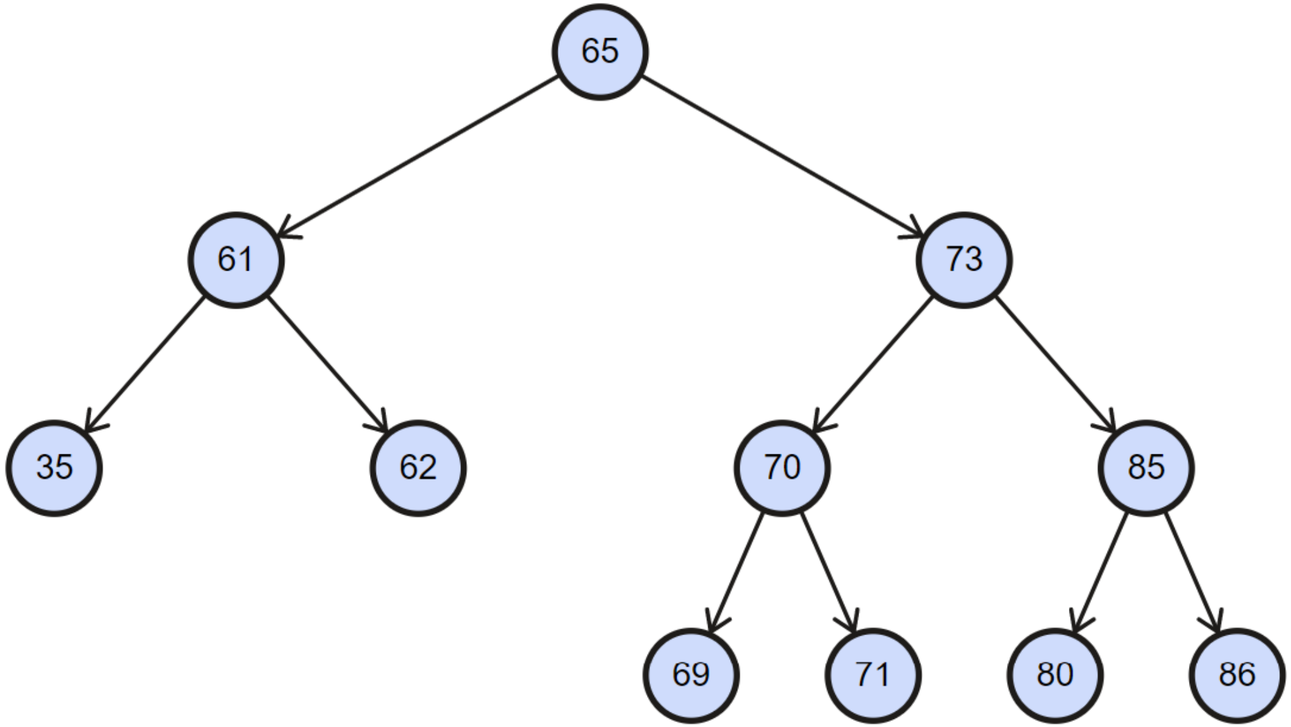
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80,



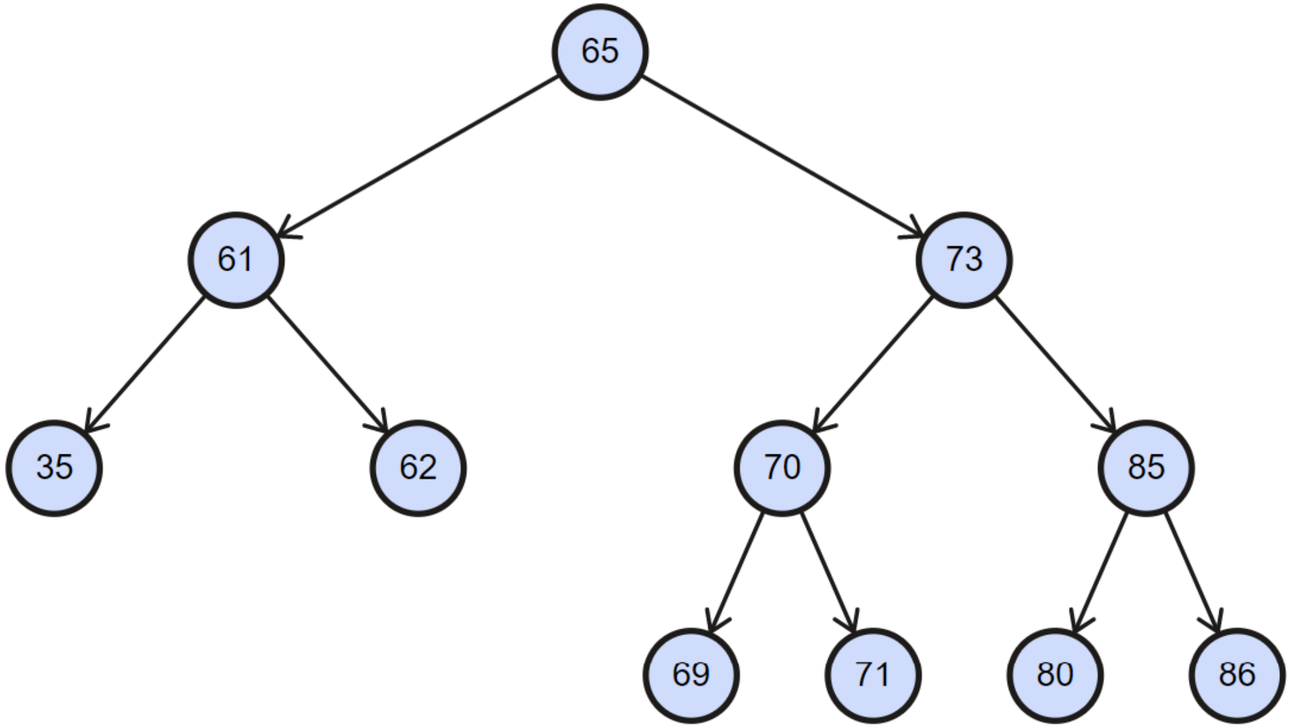
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85,



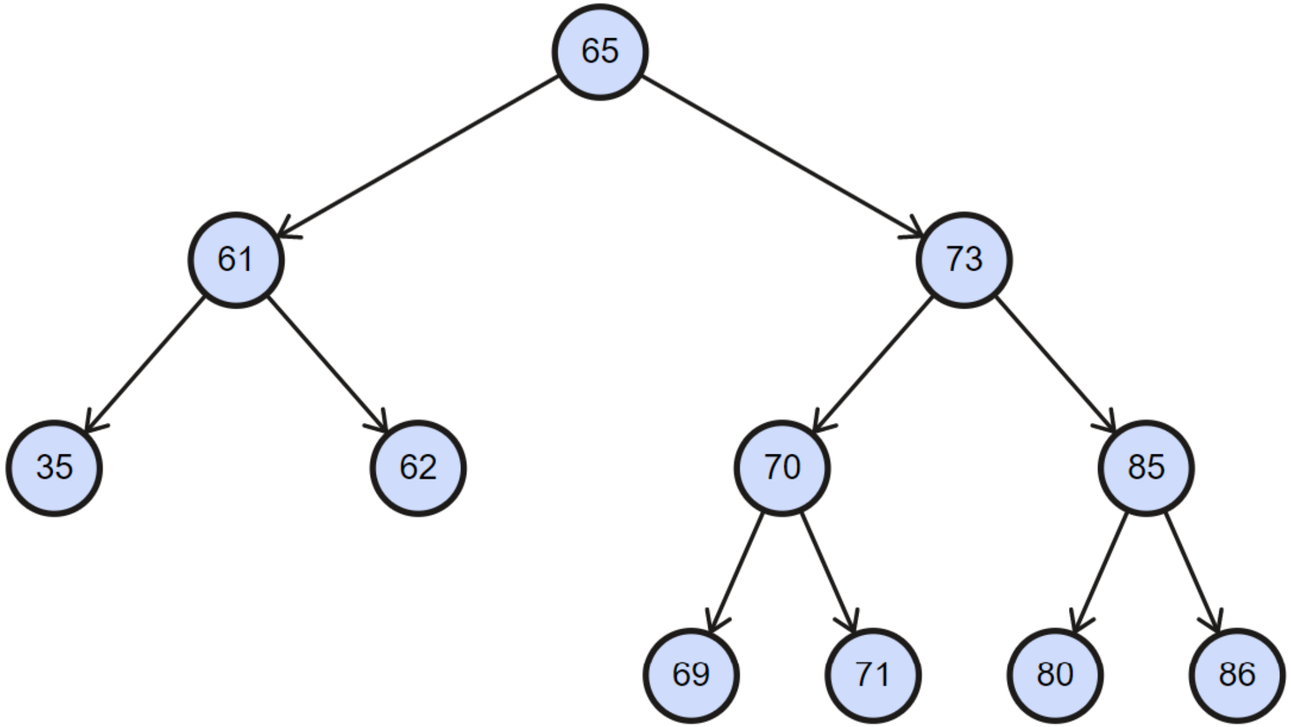
Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86



Arbori Binari de Căutare

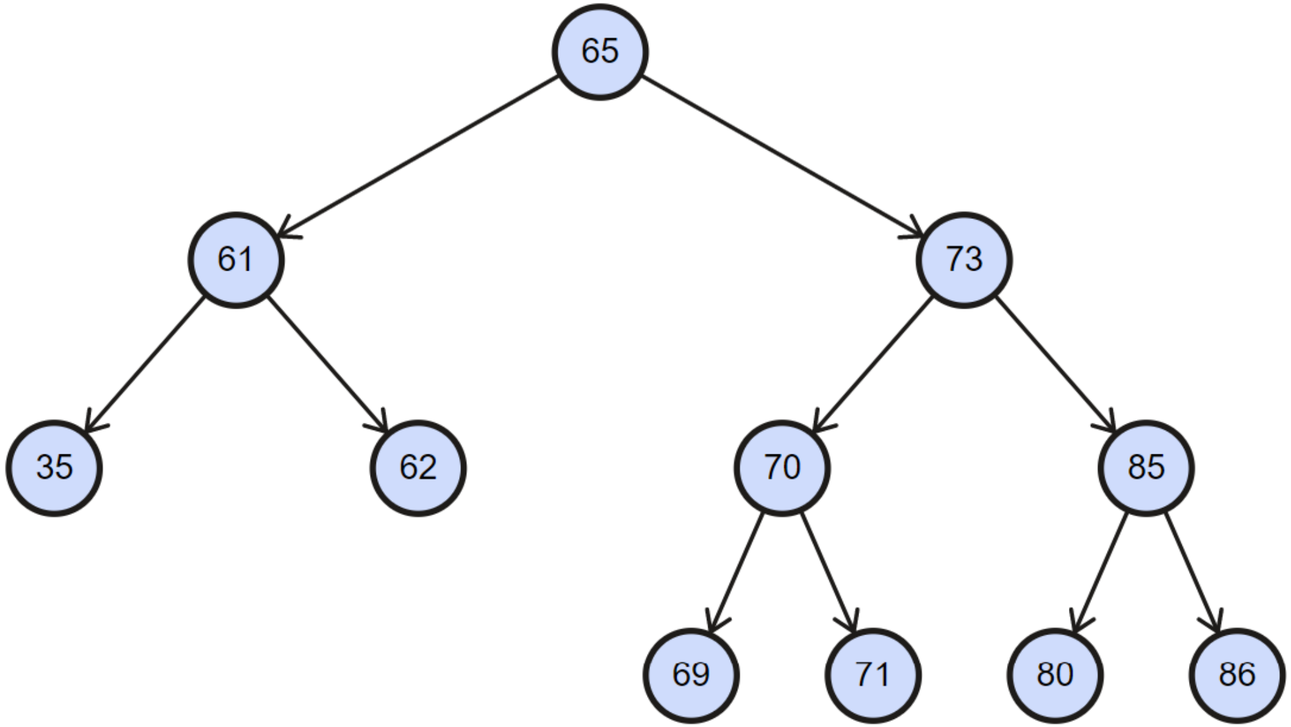
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine:



Arbori Binari de Căutare

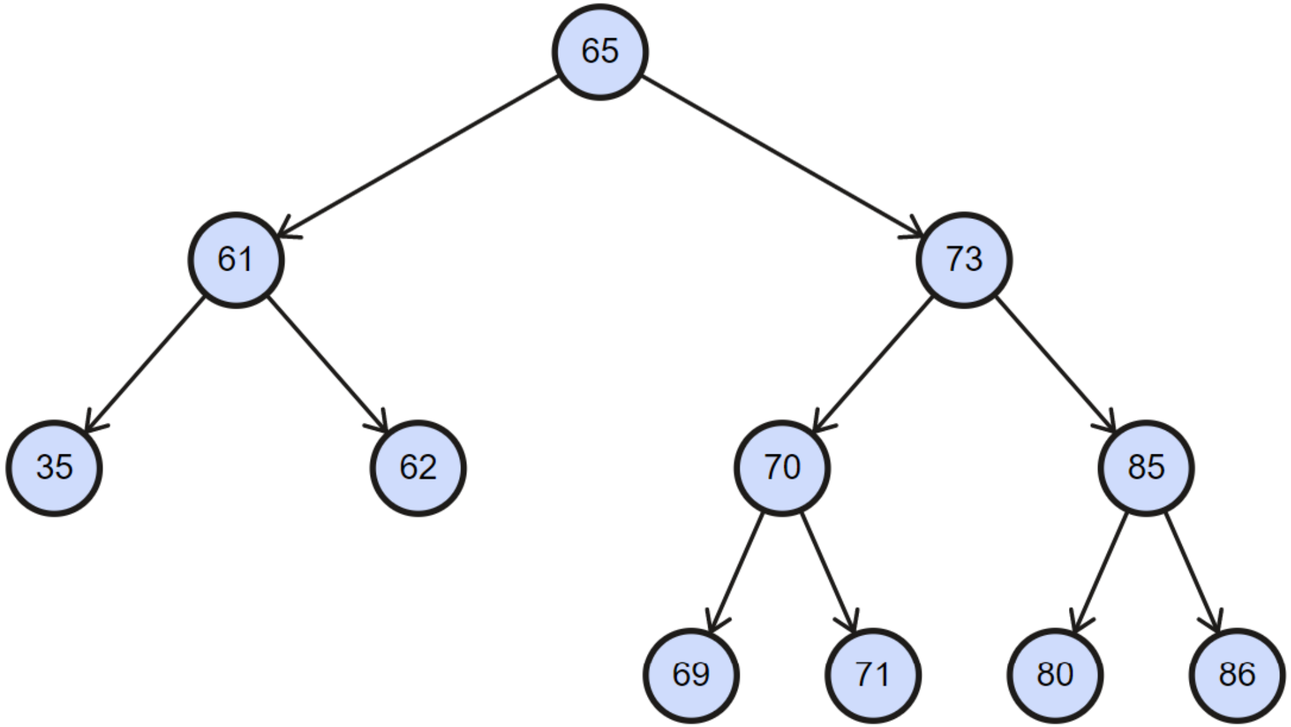
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35,



Arbori Binari de Căutare

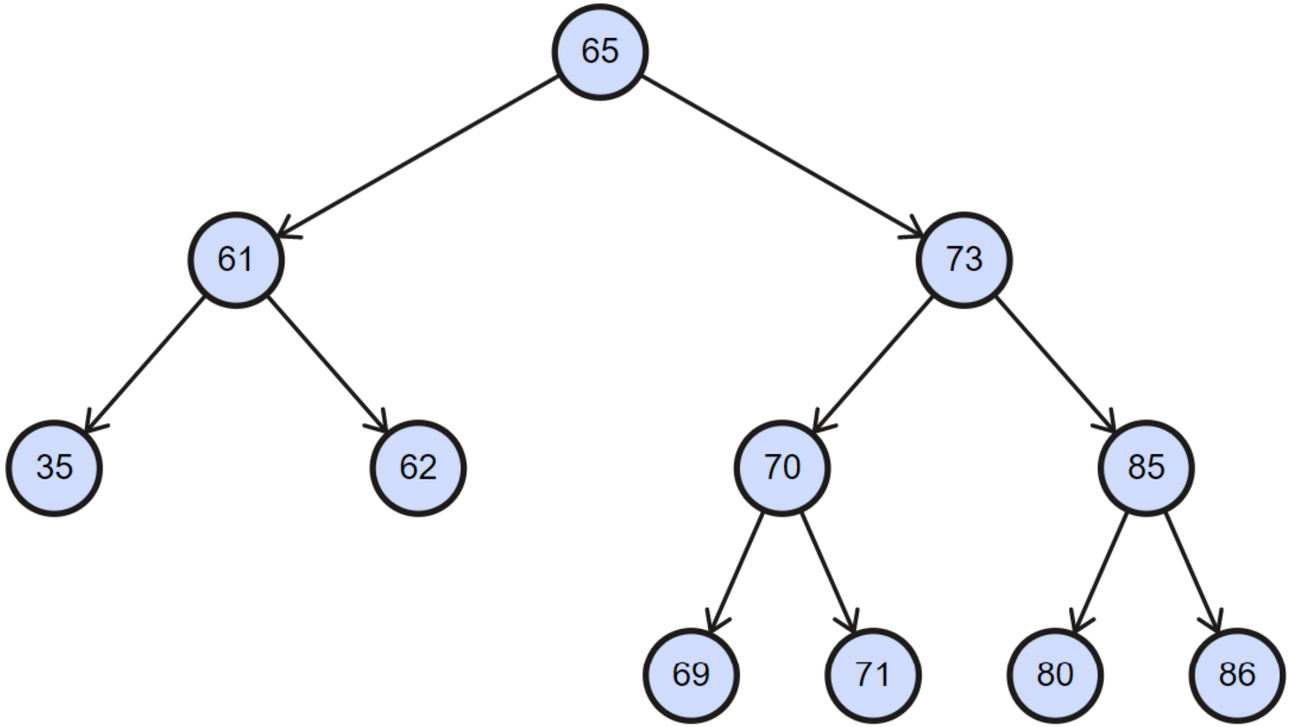
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62,



Arbori Binari de Căutare

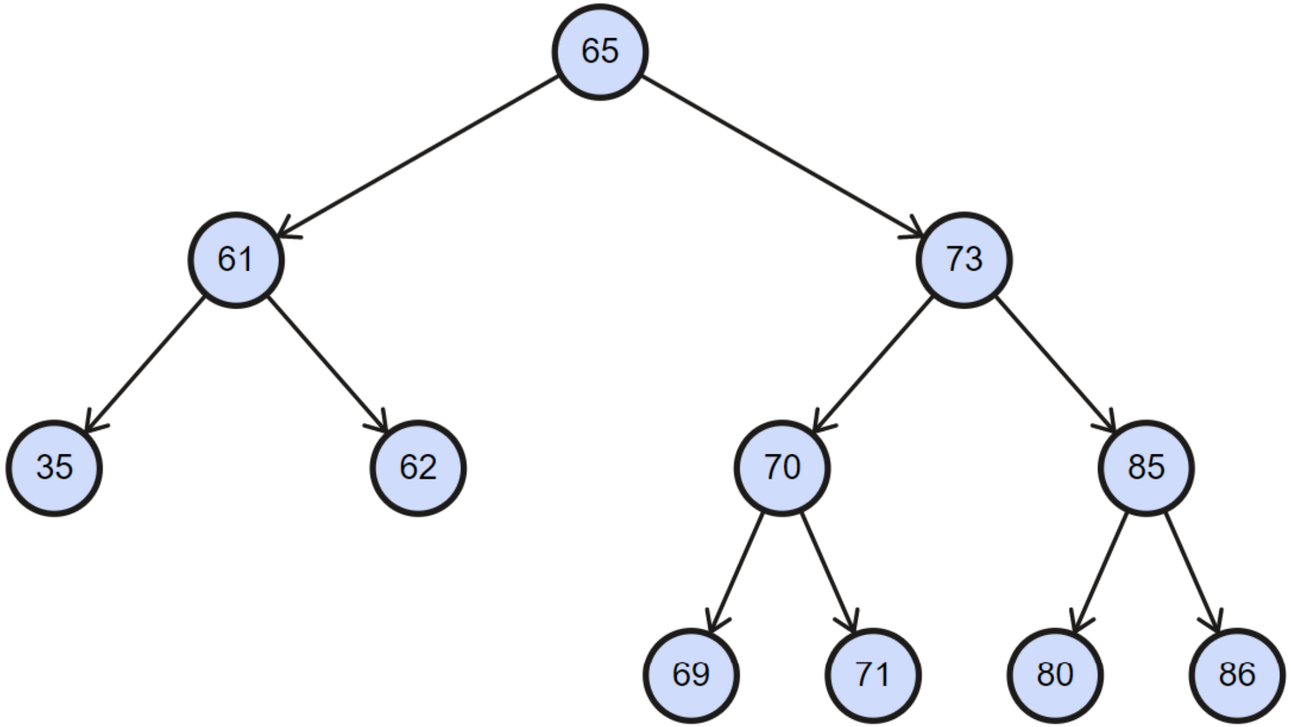
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61,



Arbori Binari de Căutare

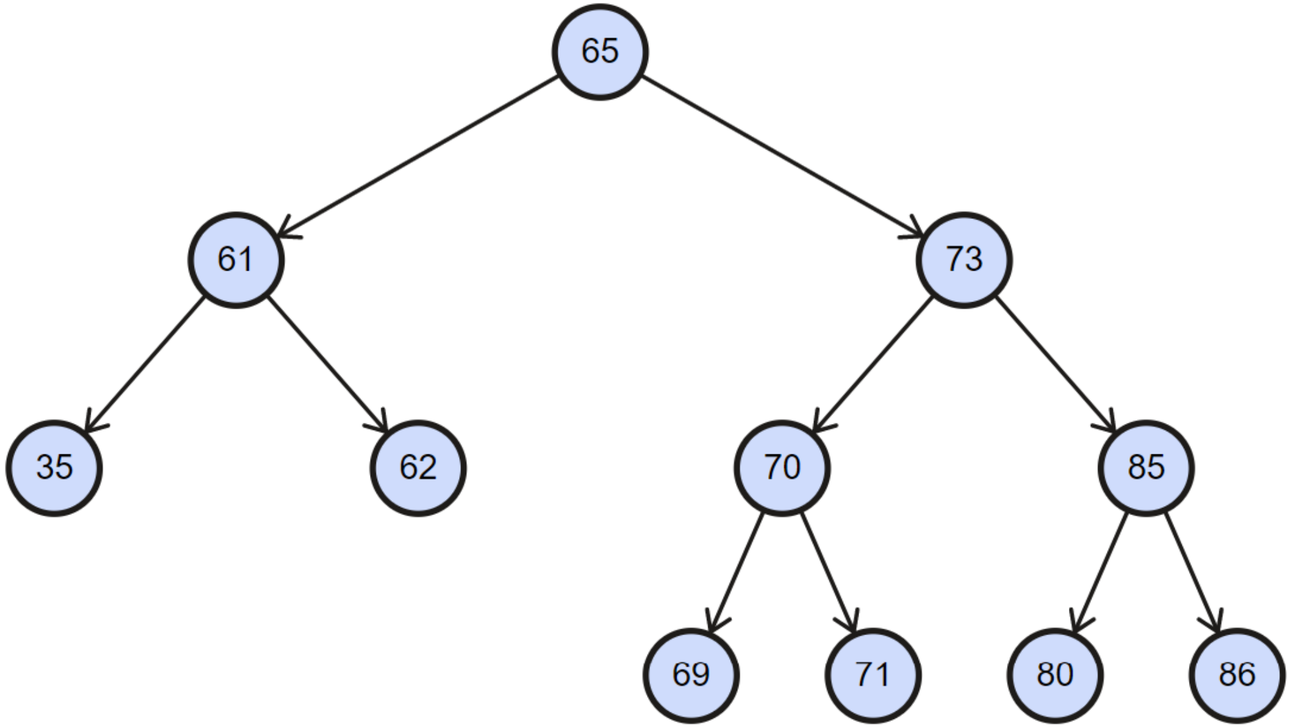
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69,



Arbori Binari de Căutare

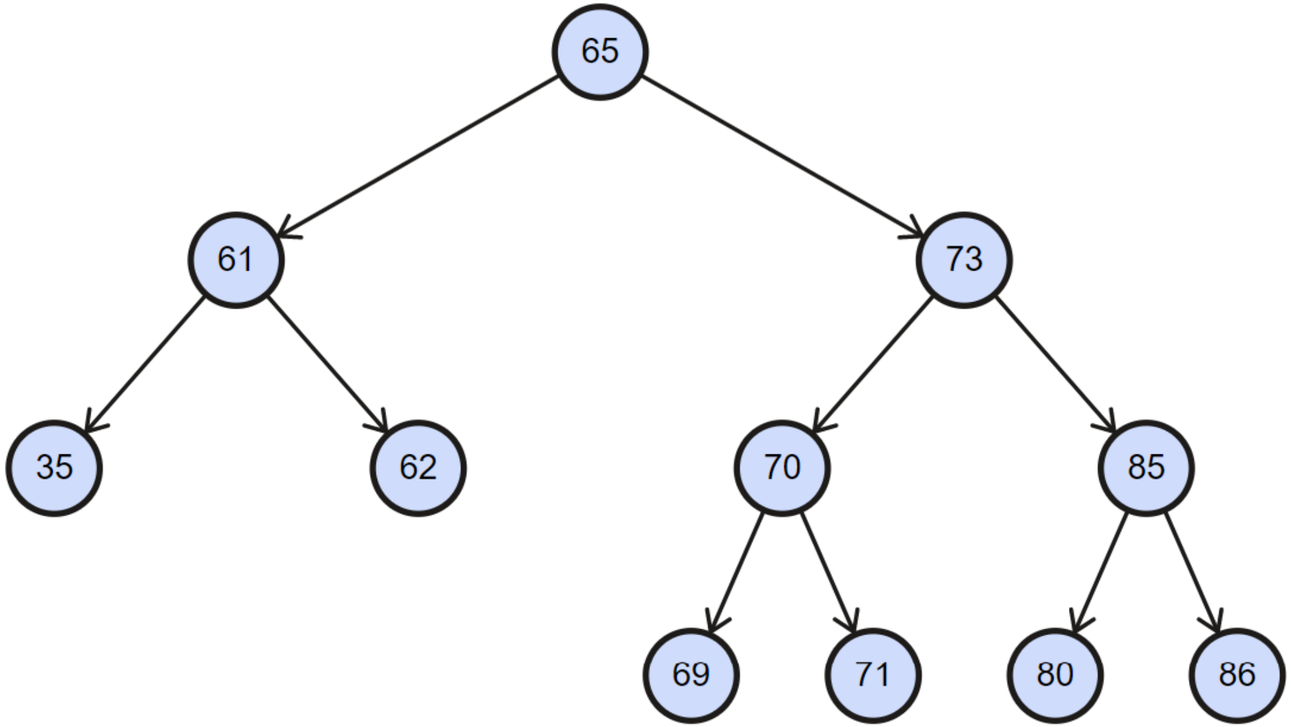
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71,



Arbori Binari de Căutare

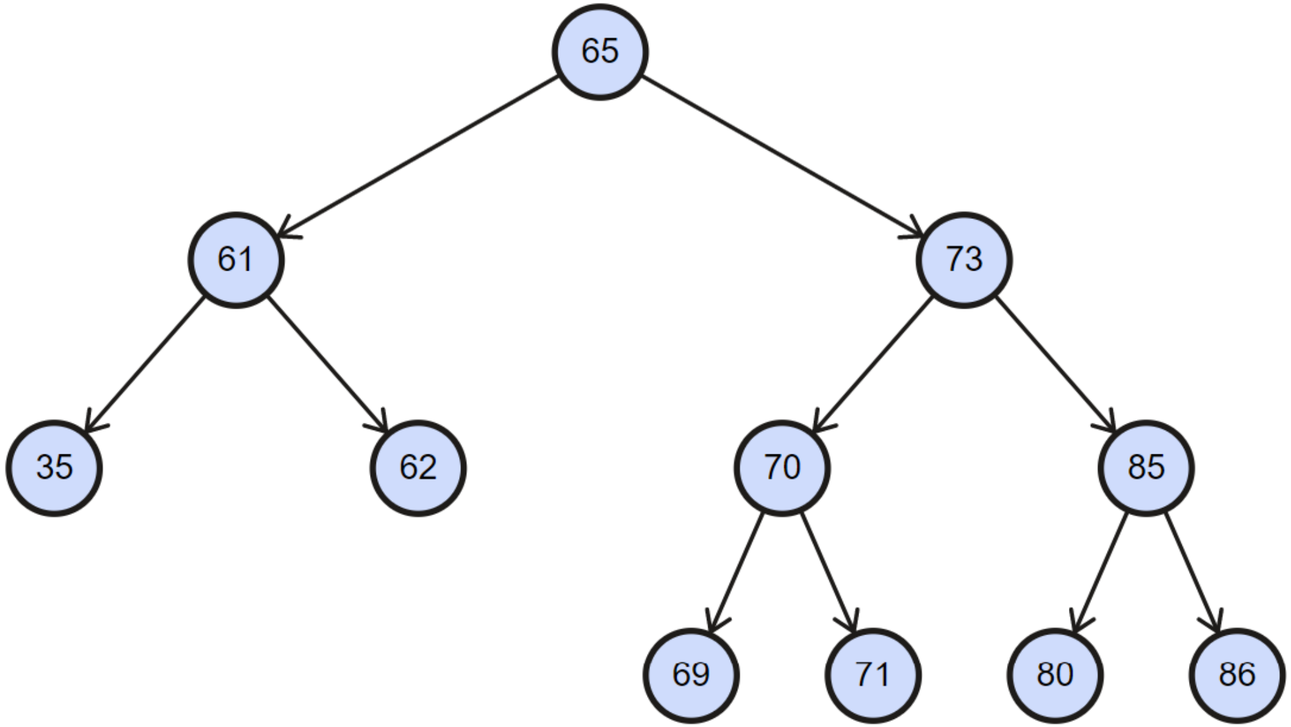
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70,



Arbori Binari de Căutare

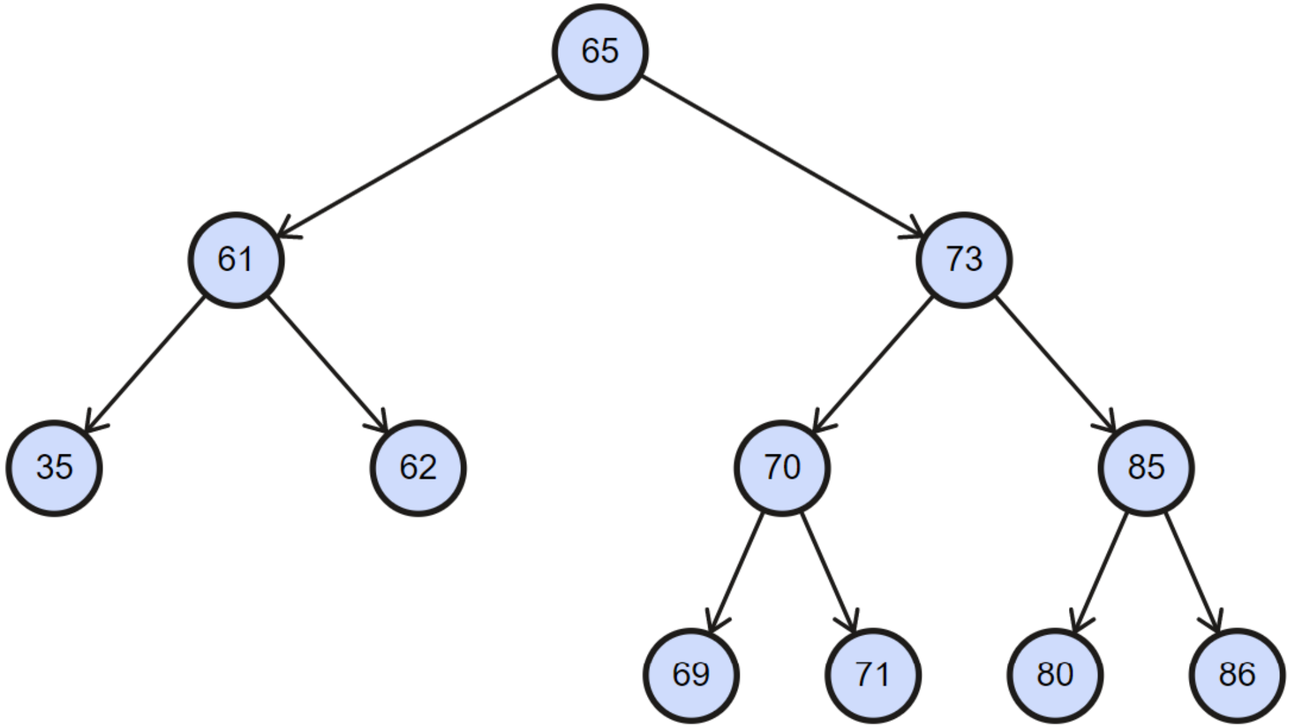
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70, 80,



Arbori Binari de Căutare

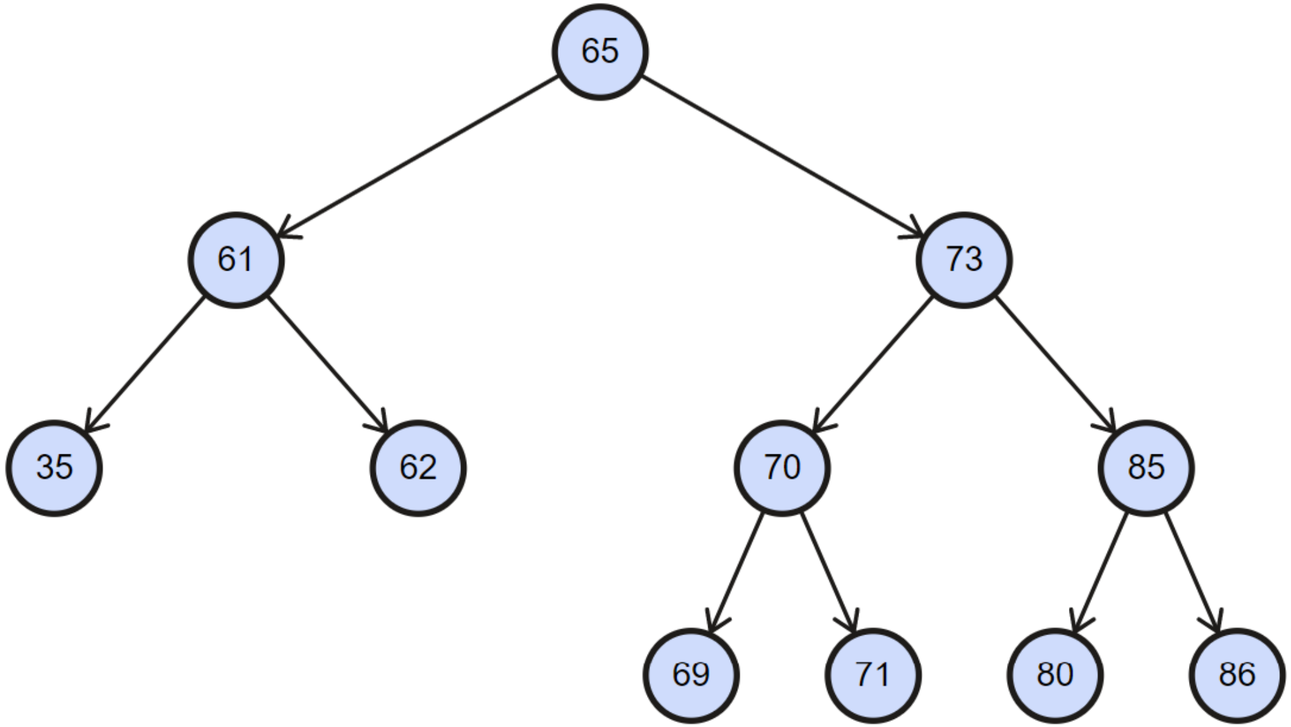
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70, 80, 86,



Arbori Binari de Căutare

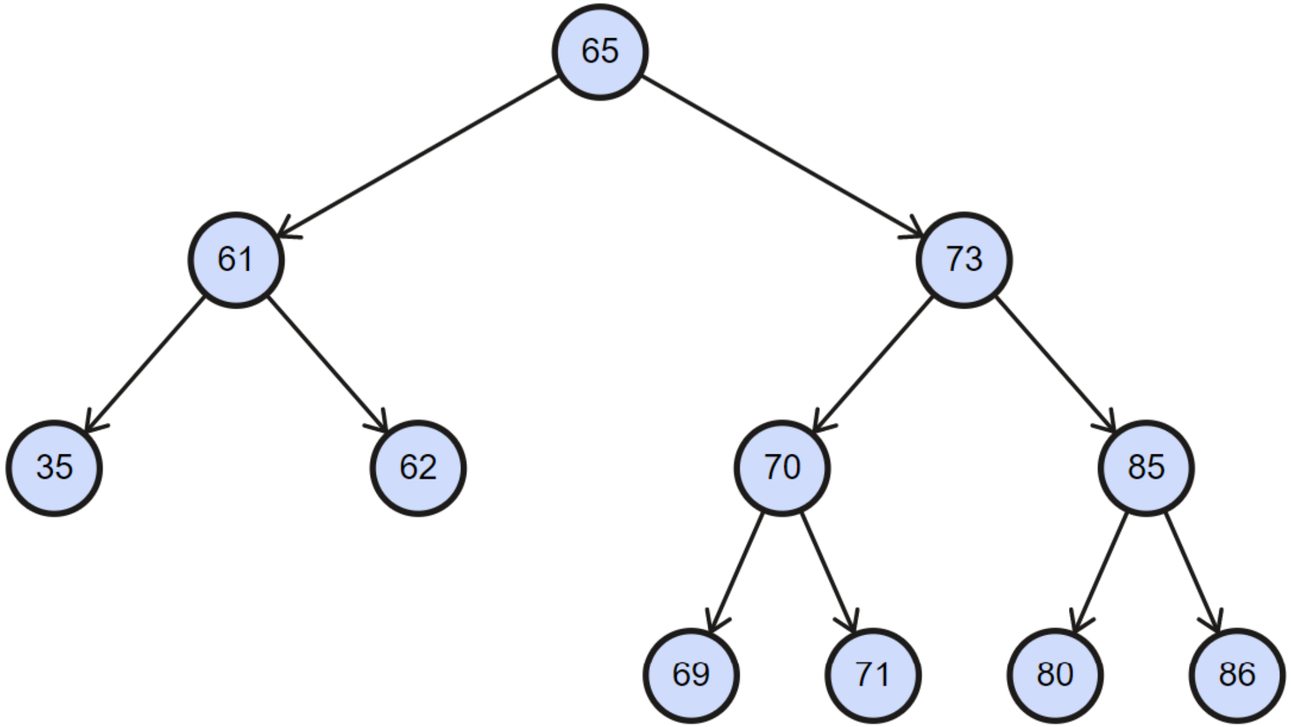
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70, 80, 86, 85,



Arbori Binari de Căutare

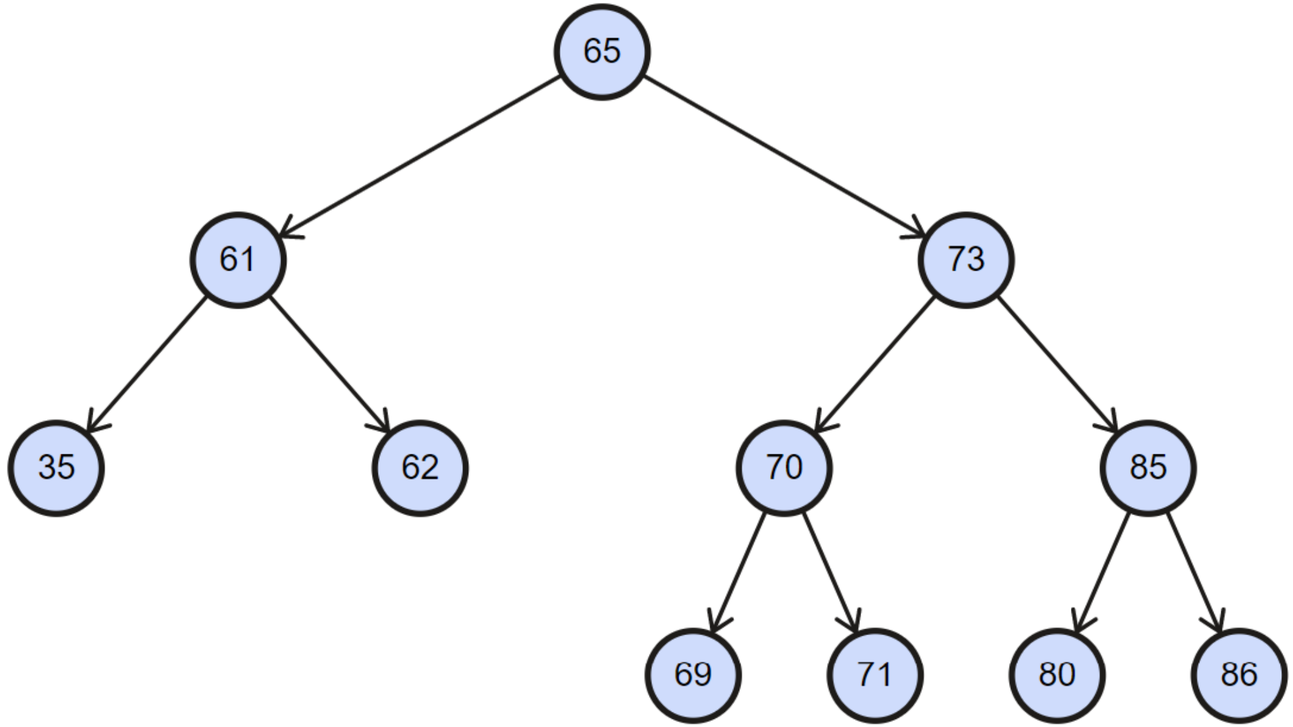
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70, 80, 86, 85, 73,



Arbori Binari de Căutare

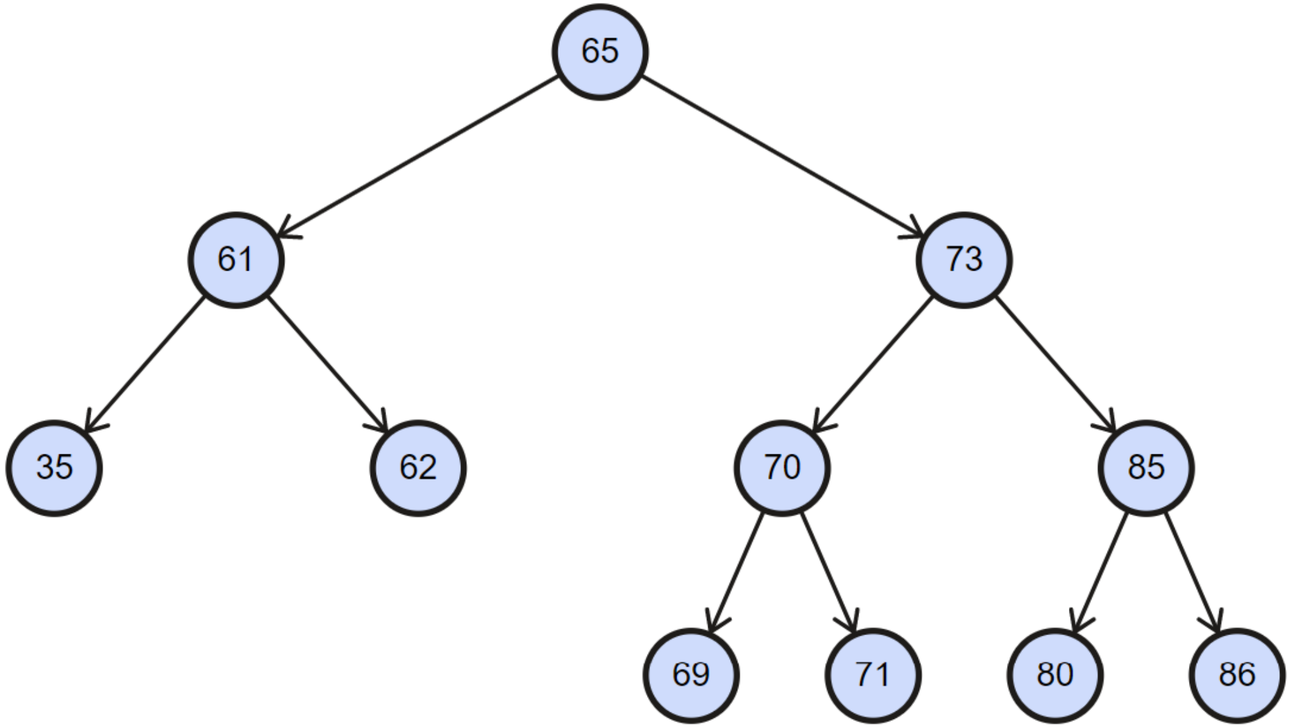
Operații uzuale pentru arbori binari de căutare:

2. Parcurgere depth-first search

Preordine : 65, 61, 35, 62, 73, 70, 69, 71, 85, 80, 86

Inordine : 35, 61, 62, 65, 69, 70, 71, 73, 80, 85, 86

Postordine: 35, 62, 61, 69, 71, 70, 80, 86, 85, 73, 65

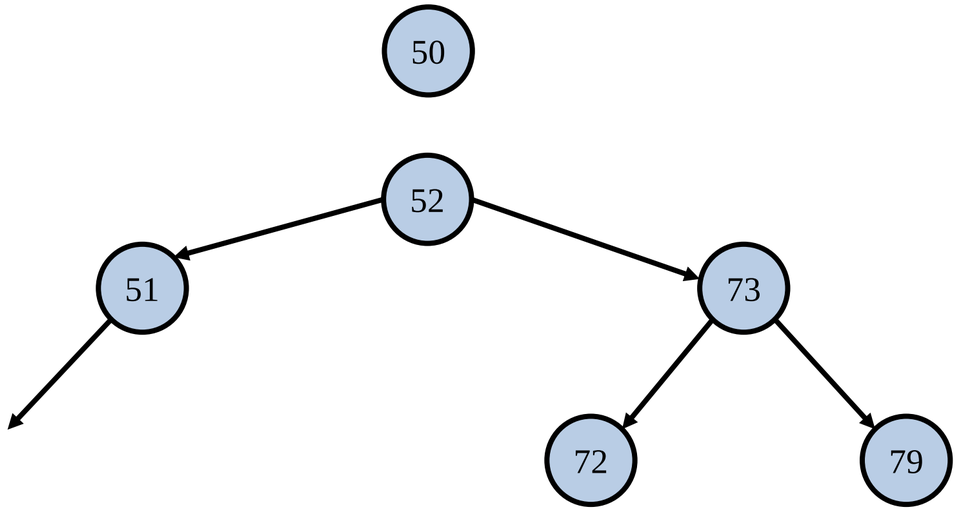


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

Inserare

1. Alocare memorie arbore/nod;
2. Inițializare atât cheia, cât și pointerii nodurilor stânga și dreapta cu valoarea dorită și respectiv cu NULL;
3. Dacă nodul ce va fi inserat este primul nod din arbore, atunci nodurile stânga și dreapta indică NULL;
4. Altfel, se verifică dacă valaorea cheii nodului este mai mică decât valoarea cheii rădăcinii arborelui. Dacă testul condițional este adevărat, atunci se repetă recursiv această operație cu partea stânga a rădăcinii;
5. Dacă testul condițional este fals, se repetă operația recursiv cu partea dreapta a subarborelui rădăcinii.

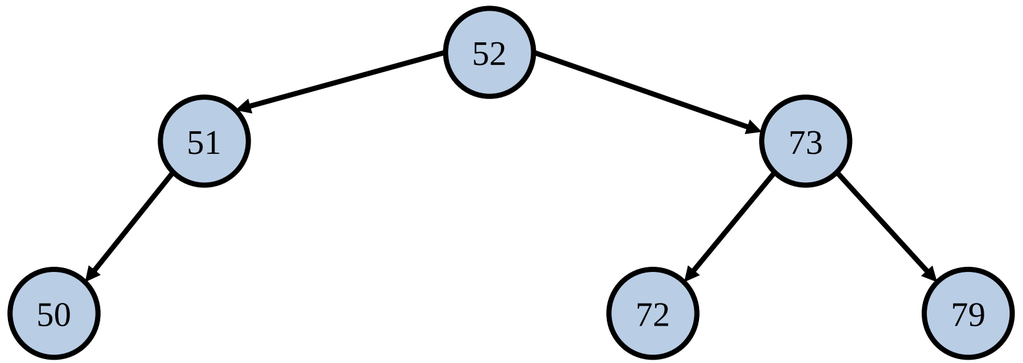


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

Ștergere

1. **Nodul ce urmează a fi șters este nod terminal**
   Se elimină nodul respectiv;

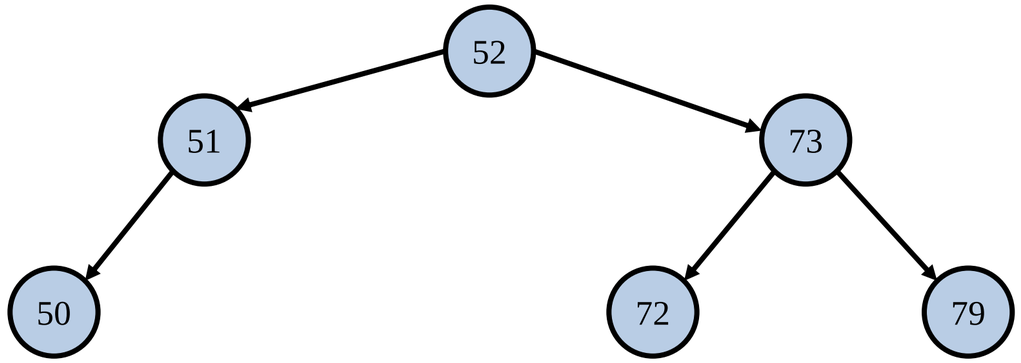


Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

Ștergere

2. **Nodul ce urmează a fi șters are doar un descendent**
   Fiul nodului ce va fi șters este legat de tatăl nodului în cauză.



Arbori Binari de Căutare

Operații uzuale pentru arbori binari de căutare:

Ștergere

3. **Nodul ce urmează a fi șters are doi descendenți**

   Se caută valoarea minimă în subarborele drept, apoi se înlocuiește cheia nodului ce va fi eliminat cu valoarea minimă găsită. Se repetă algoritmul recursiv pe ramura dreaptă, pentru a șterge nodul vizat.