Binární vyhledávací stromy I

doc. Mgr. Jiří Dvorský, Ph.D.

Katedra informatiky
Fakulta elektrotechniky a informatiky
VŠB – TU Ostrava

Prezentace ke dni 12. února 2020



Osnova přednášky

Binární vyhledávací stromy I

- Binární stromy
- Binární vyhledávací stromy

Binární strom

Definice

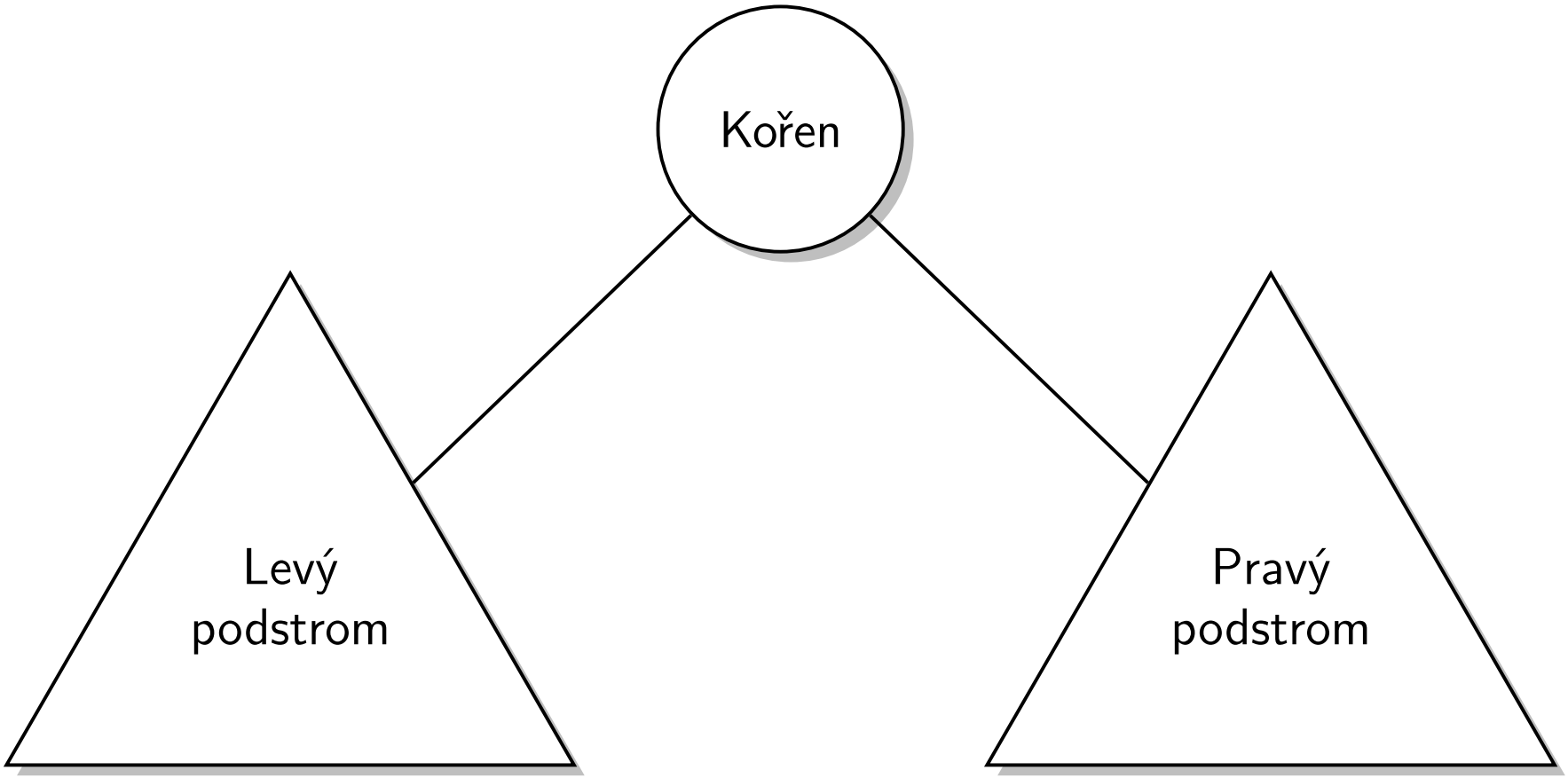
Binární strom je struktura definovaná nad konečnou množnou uzlů, která:

Pravidlo 1 – neobsahuje žádný uzel nebo

Pravidlo 2 – je složena ze tří disjunktních množin uzlů:

- kořene stromu,
- binárního stromu zvaného levý podstrom,
- a binárního stromu zvaného pravý podstrom.

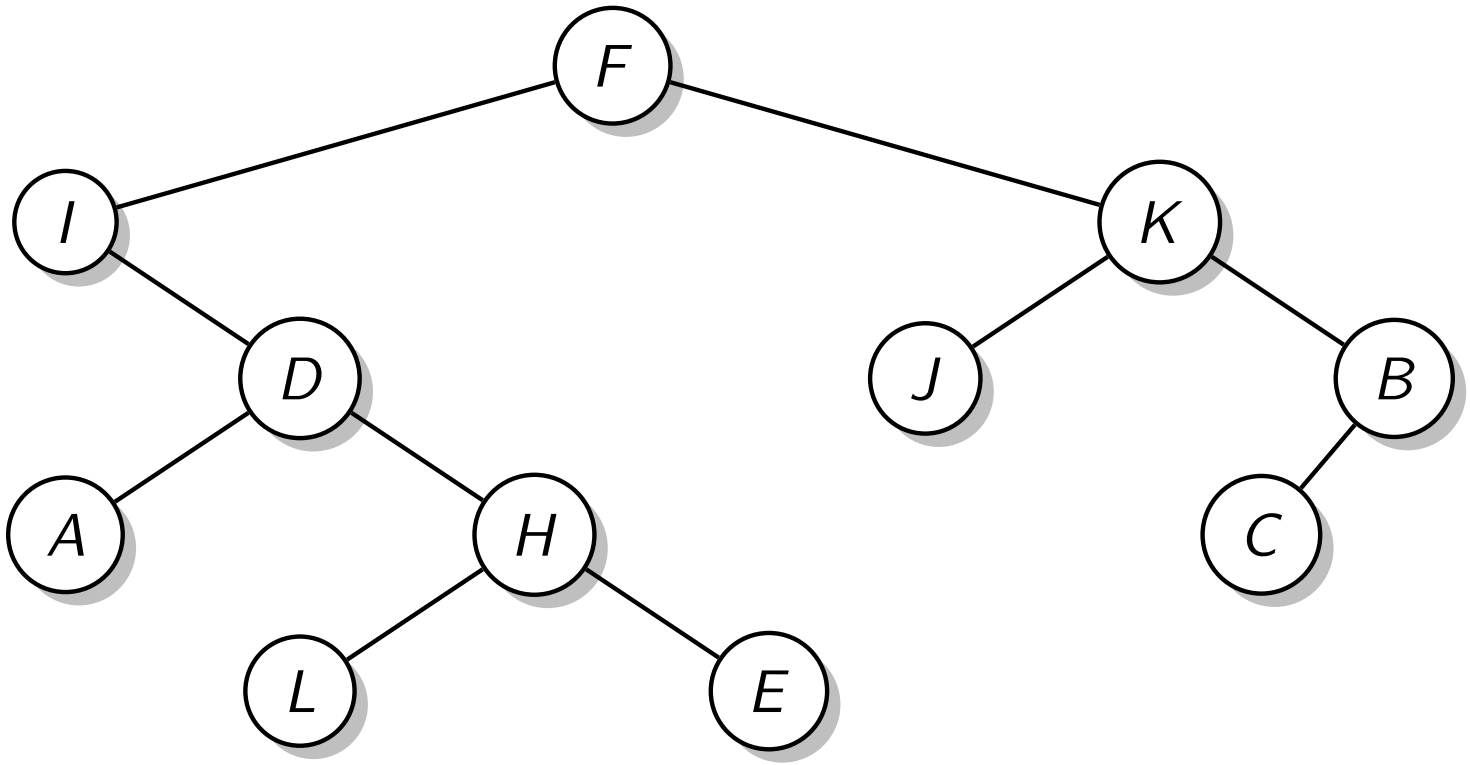
Binární strom – grafické znázornění rekurzivní definice



Binární vs. seřazený strom

- Binární strom není seřazený strom, kde má každý uzel stupeň nejvýše dva.
- Pokud má uzel jednoho potomka, je rozdíl zda je to potomek levý nebo pravý důležitý.
- V seřazeném stromu není možné rozlišit zda je jediný potomek levý nebo pravý, prostě je jeden.

Binární strom, příklad



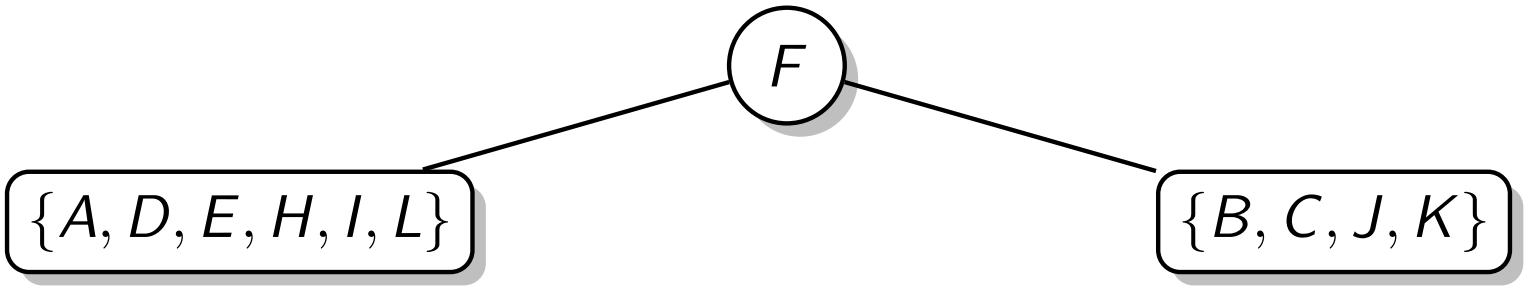
Budování binárního stromu podle rekurzivní definice

- Výchozí množina uzlů: $\{A, B, C, D, E, F, H, I, J, K, L\}$
- Postupně budeme aplikovat pravidla z definice binárního stromu.
- Uzly ve tvaru kruhu odpovídají již vytvořeným uzlům binárního stromu, viz Pravidlo 2.
- Uzly ve formě množin odpovídají dosud nevytvořeným podstromům, viz Pravidlo 2. Zde budeme rekurzivně pokračovat v tvorbě binárního stromu.
- Prázdné množiny odpovídají prázdným podstromům, viz Pravidlo 1.

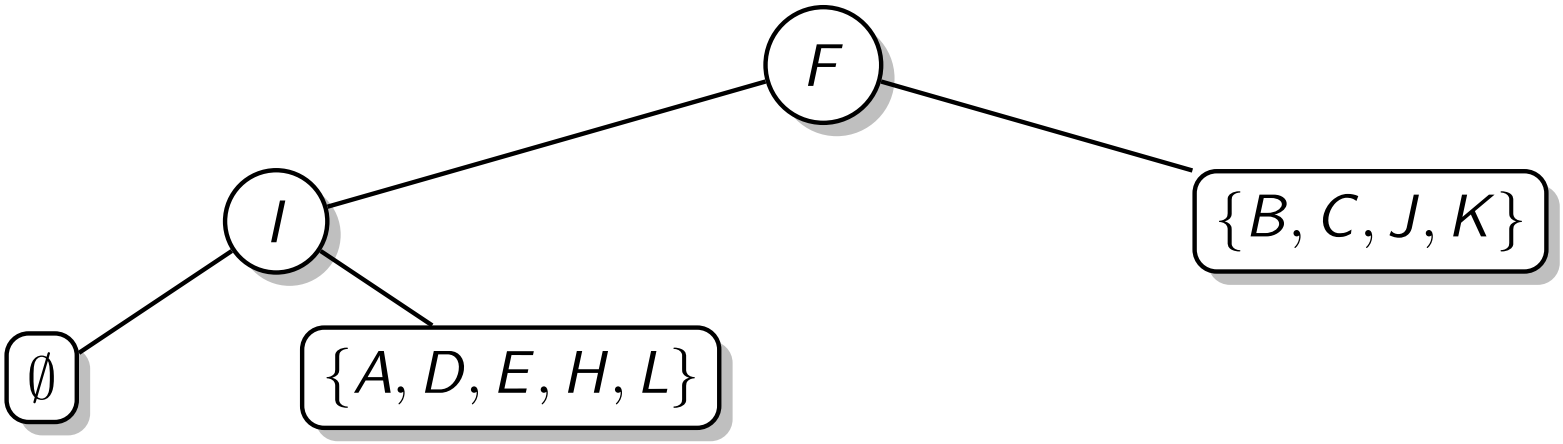
Budování binárního stromu podle rekurzivní definice

$$\{A, B, C, D, E, F, H, I, J, K, L\}$$

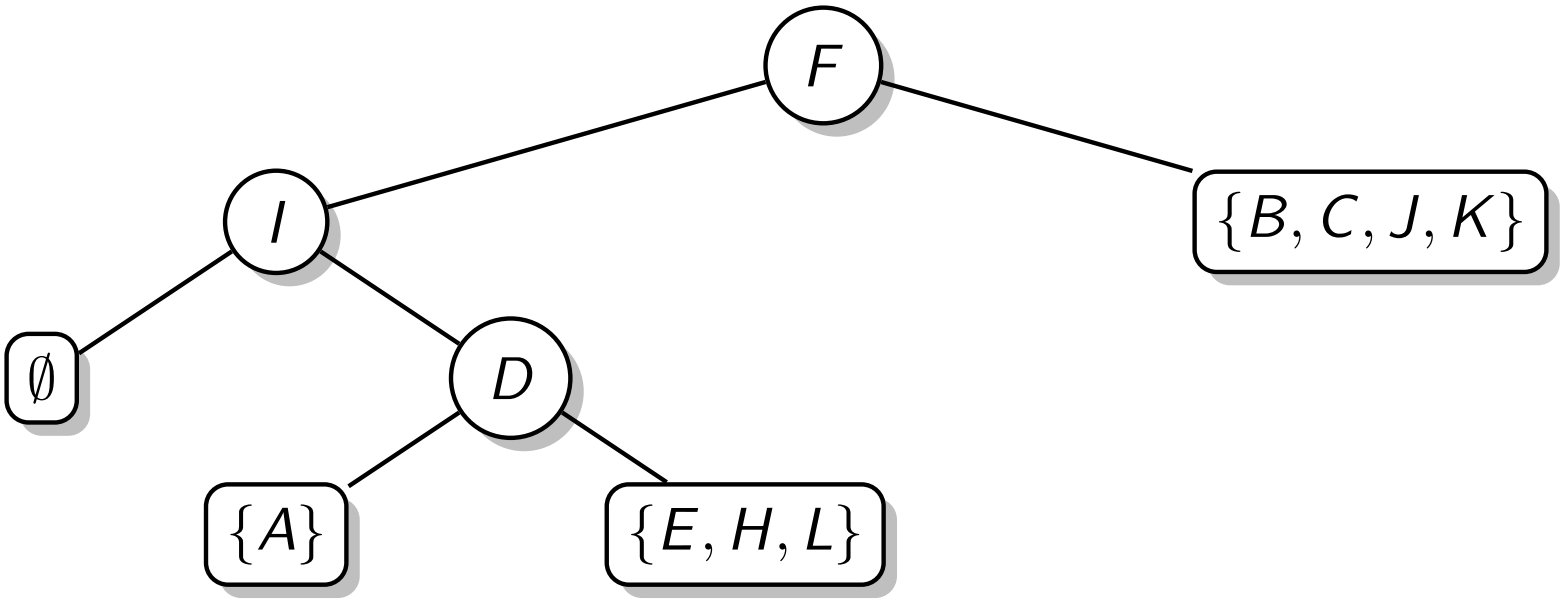
Budování binárního stromu podle rekurzivní definice



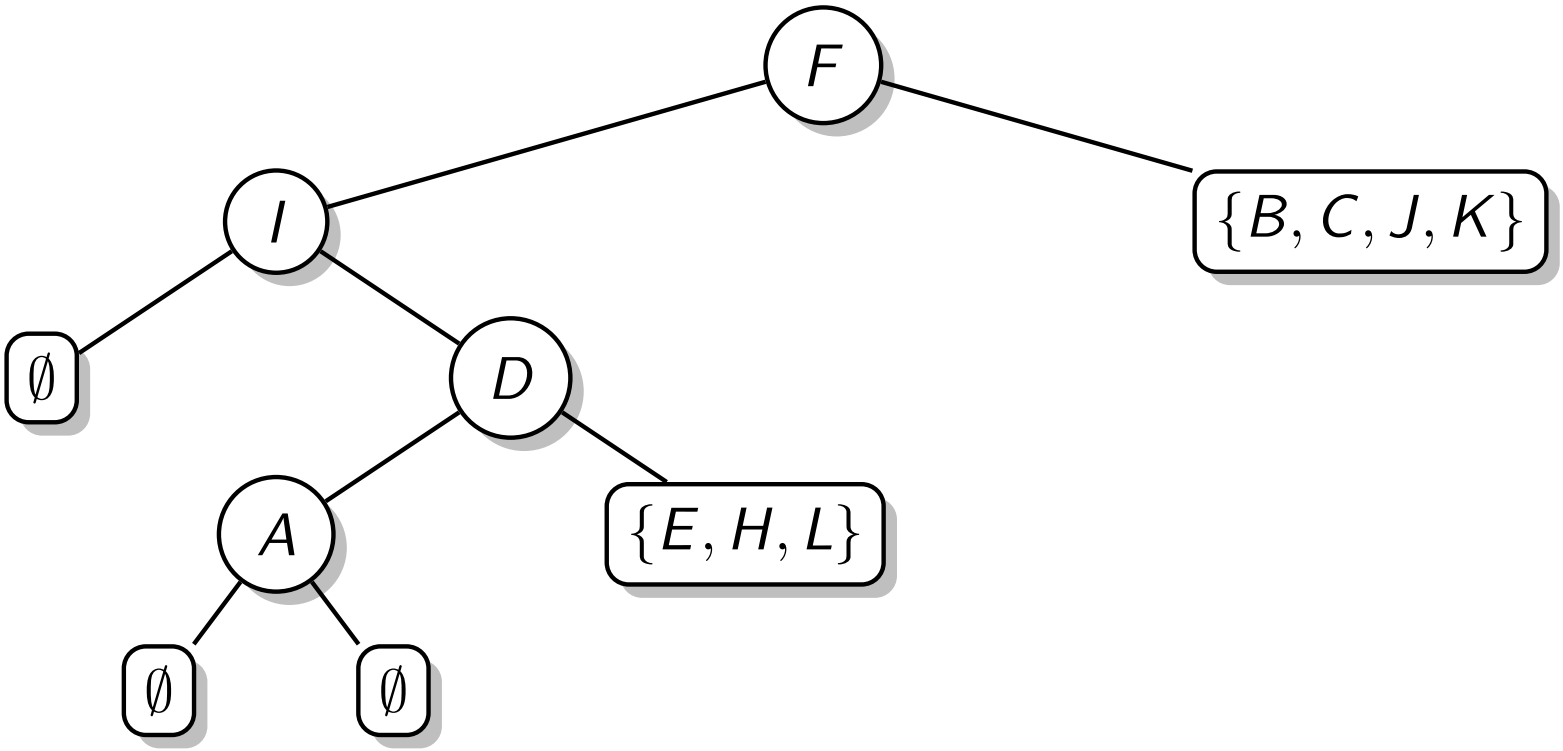
Budování binárního stromu podle rekurzivní definice



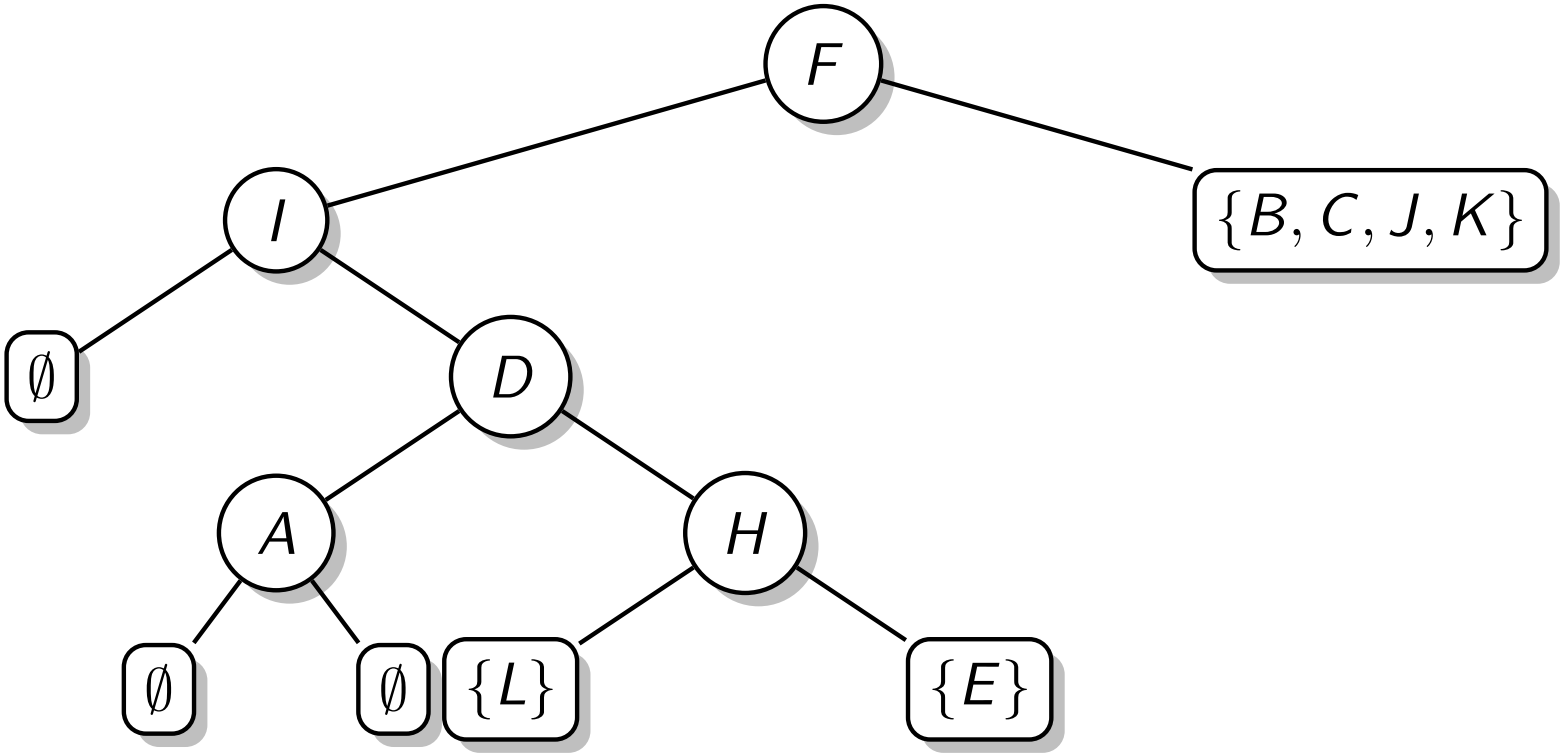
Budování binárního stromu podle rekurzivní definice



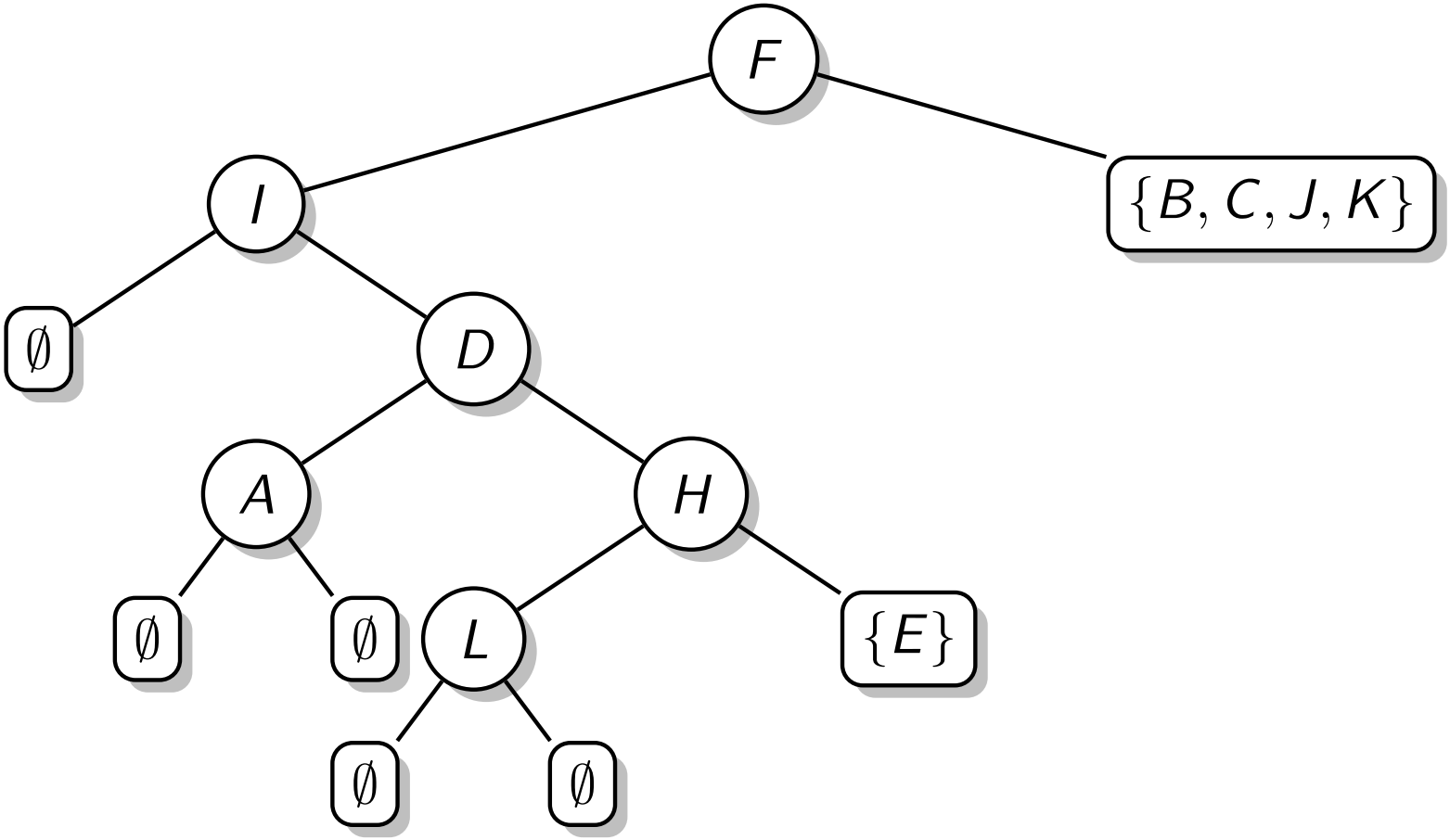
Budování binárního stromu podle rekurzivní definice



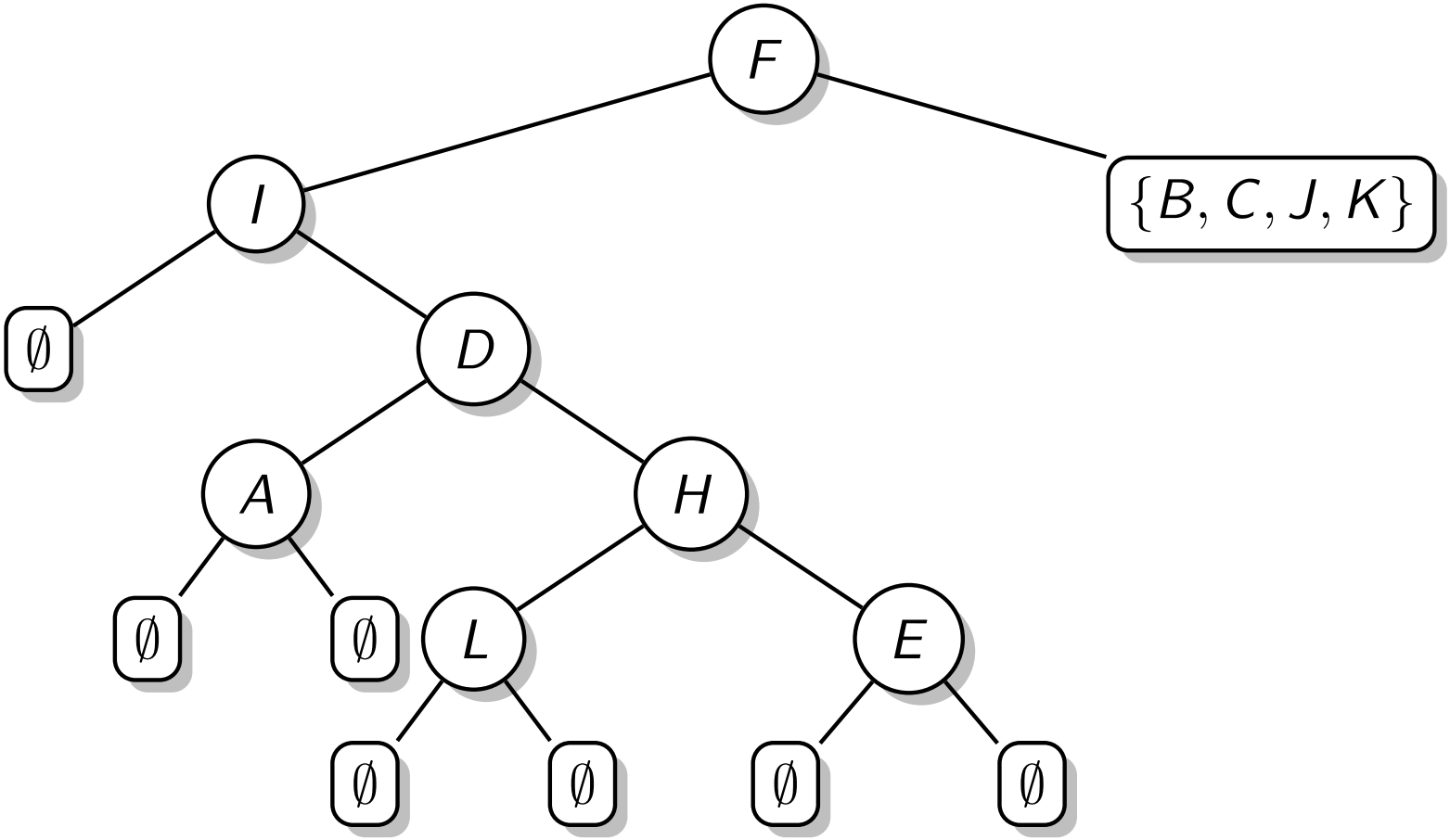
Budování binárního stromu podle rekurzivní definice



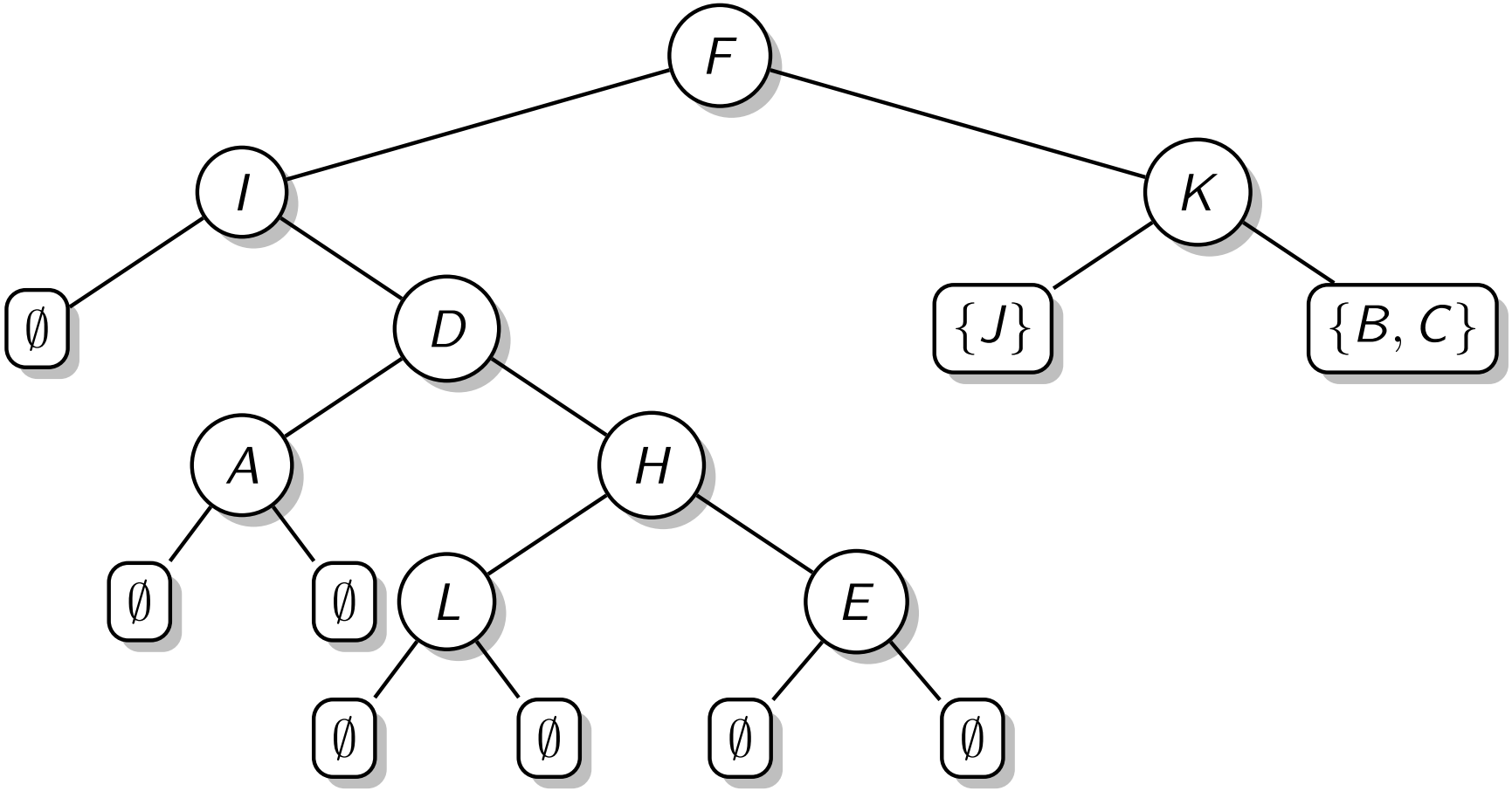
Budování binárního stromu podle rekurzivní definice



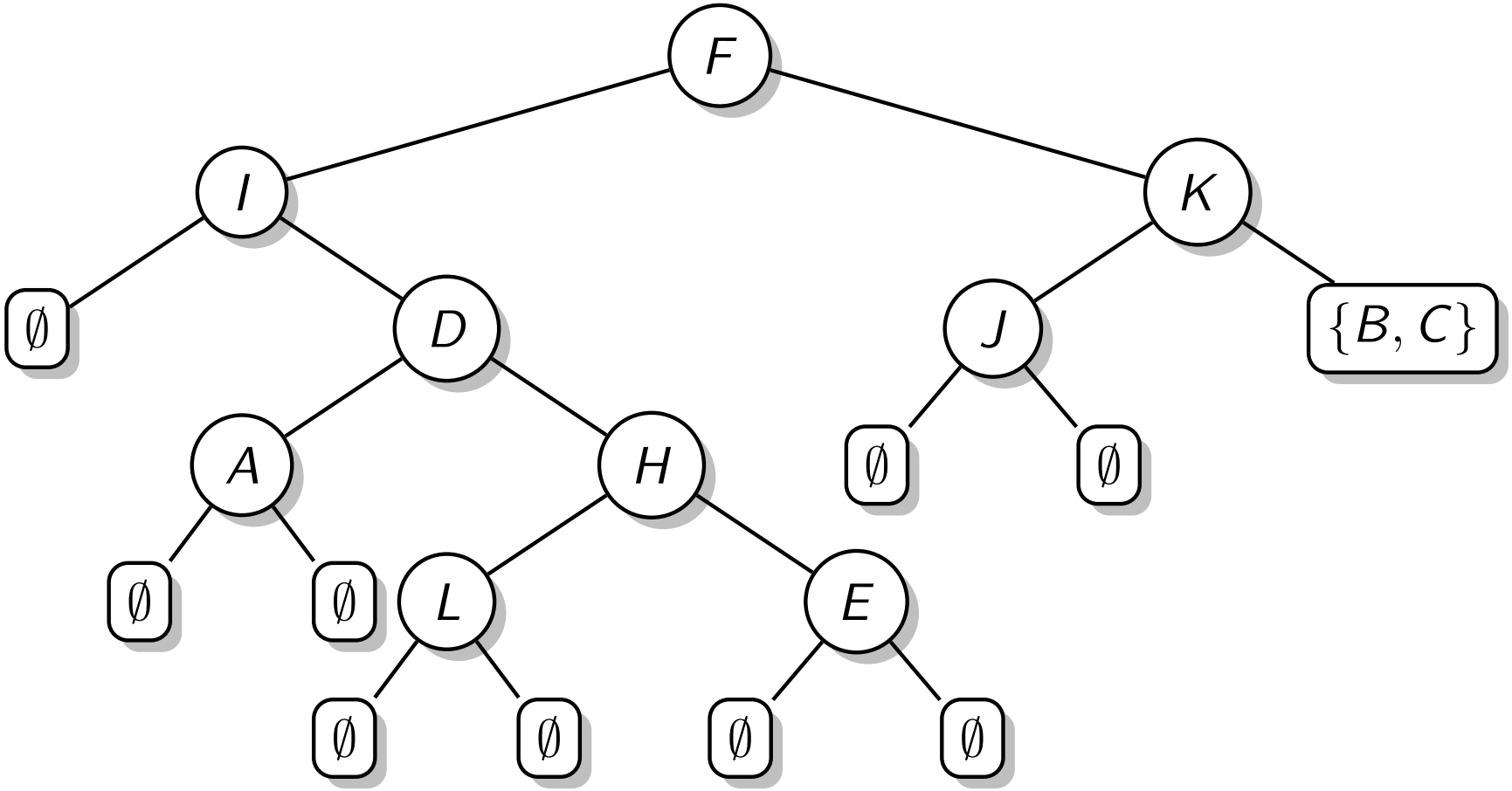
Budování binárního stromu podle rekurzivní definice



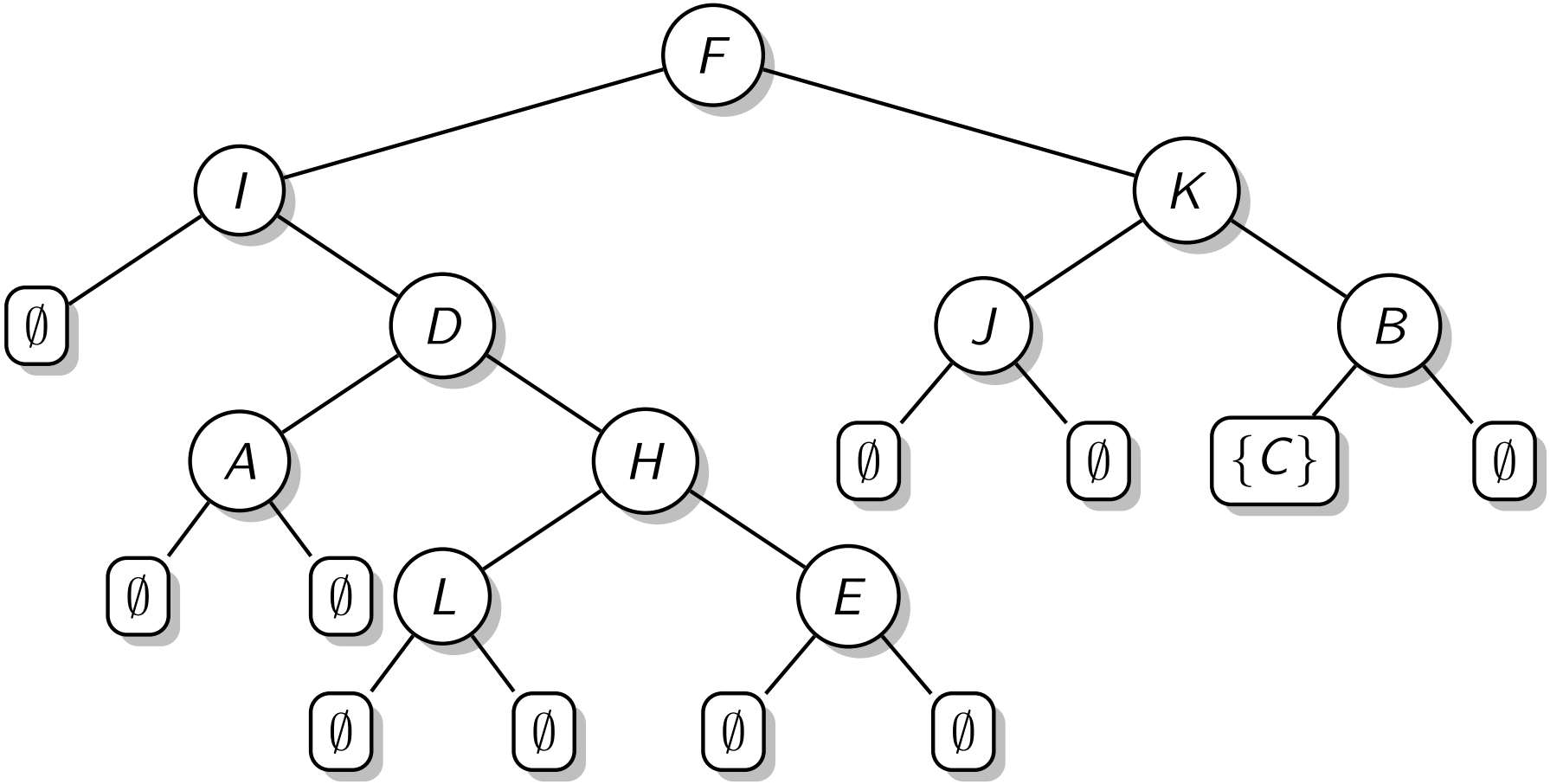
Budování binárního stromu podle rekurzivní definice



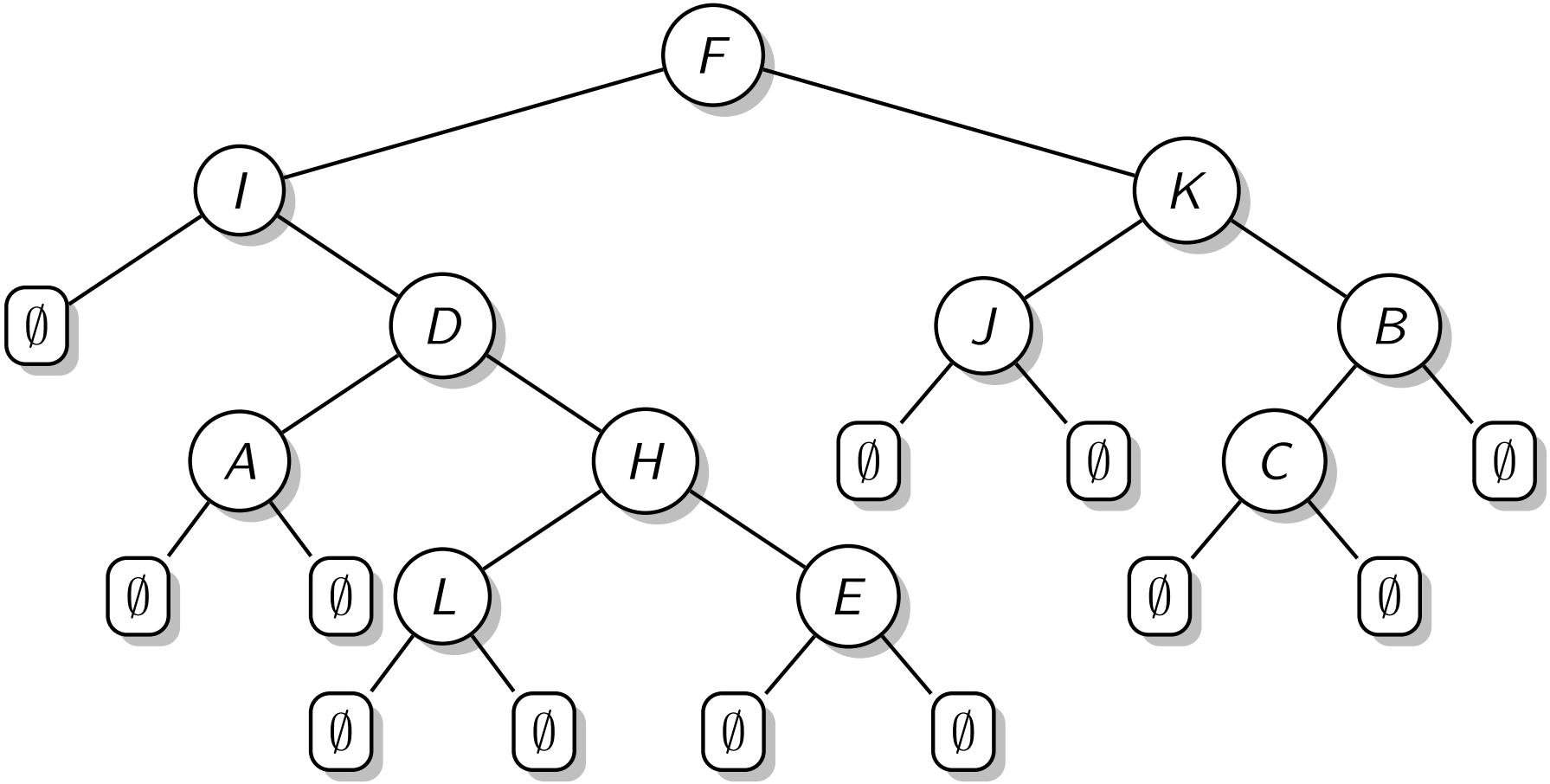
Budování binárního stromu podle rekurzivní definice



Budování binárního stromu podle rekurzivní definice

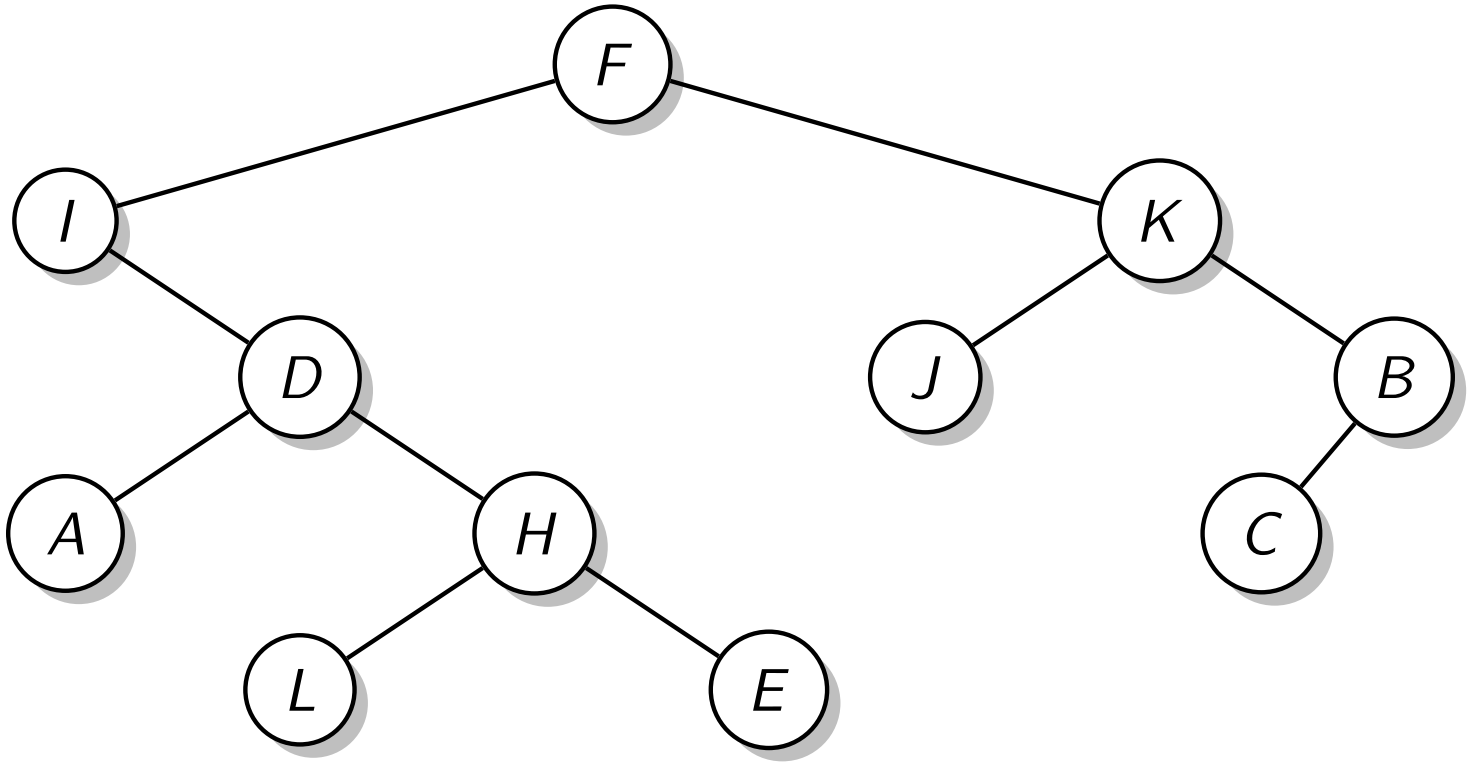


Budování binárního stromu podle rekurzivní definice

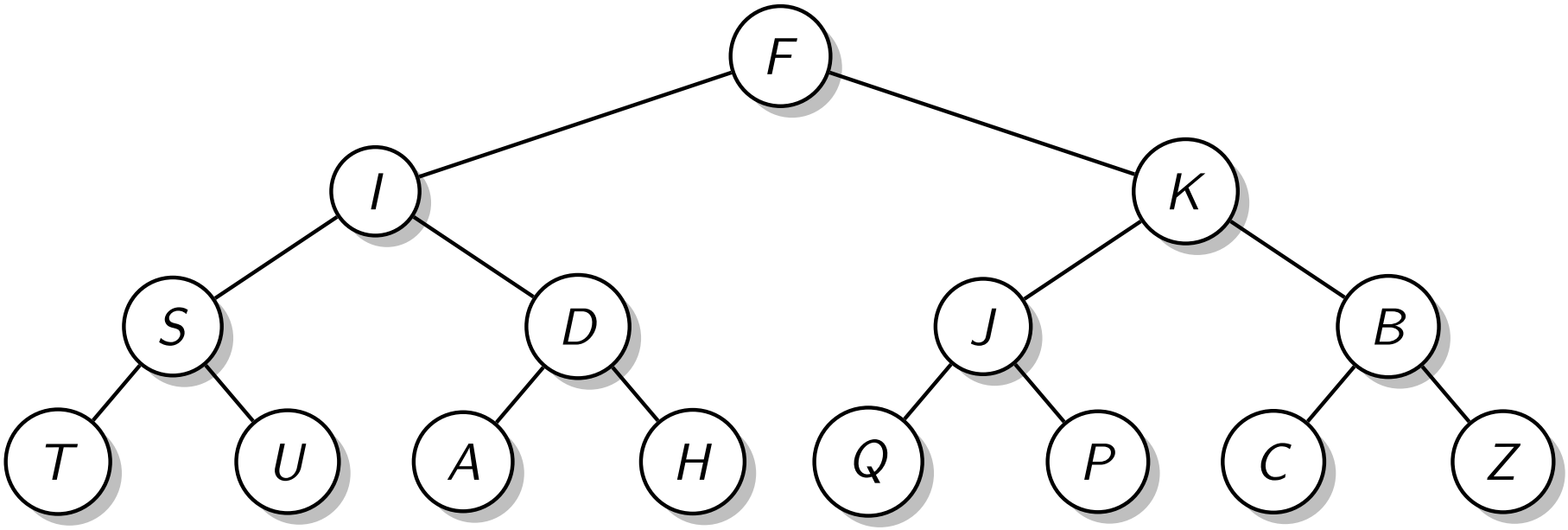


Budování binárního stromu podle rekurzivní definice

Odstraníme-li vizualizaci prázdných podstromů, zobrazených jako prázdné množiny, dostáváme výsledný binární strom v obvyklé podobě.

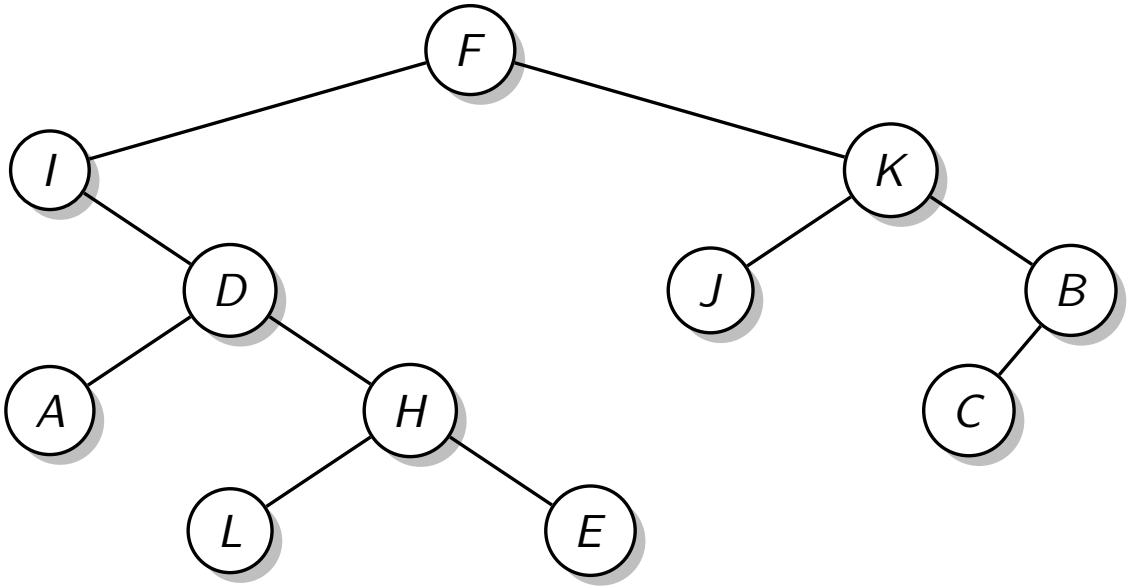


Úplný binární strom



Úplný binární strom – každý vnitřní uzel má právě dva potomky.

Binární vyhledávací strom



- Jak využít binární stromy jako datovou strukturu? Jakým způsobem v nich organizovat data?
- Libovolně? Nesmysl – jde o zbytečně komplikovaný seznam!

Řešením je využít vlastností stromu (souvislost a jedinečnost cesty z uzlu do uzlu) a doplnit je vhodným „**navigačním pravidlem**“.

Binární vyhledávací strom – „navigační pravidlo“

Nechť y je uzel v binárním stromu. Potom pro každý uzel x v levém podstromu uzlu y a každý uzel z v pravém podstromu uzlu y platí

$$x_{key} < y_{key} < z_{key}.$$

Binární strom, ve kterém pro všechny jeho uzly platí toto pravidlo nazýváme **binární vyhledávací strom** (angl. binary search tree).

Poznámky

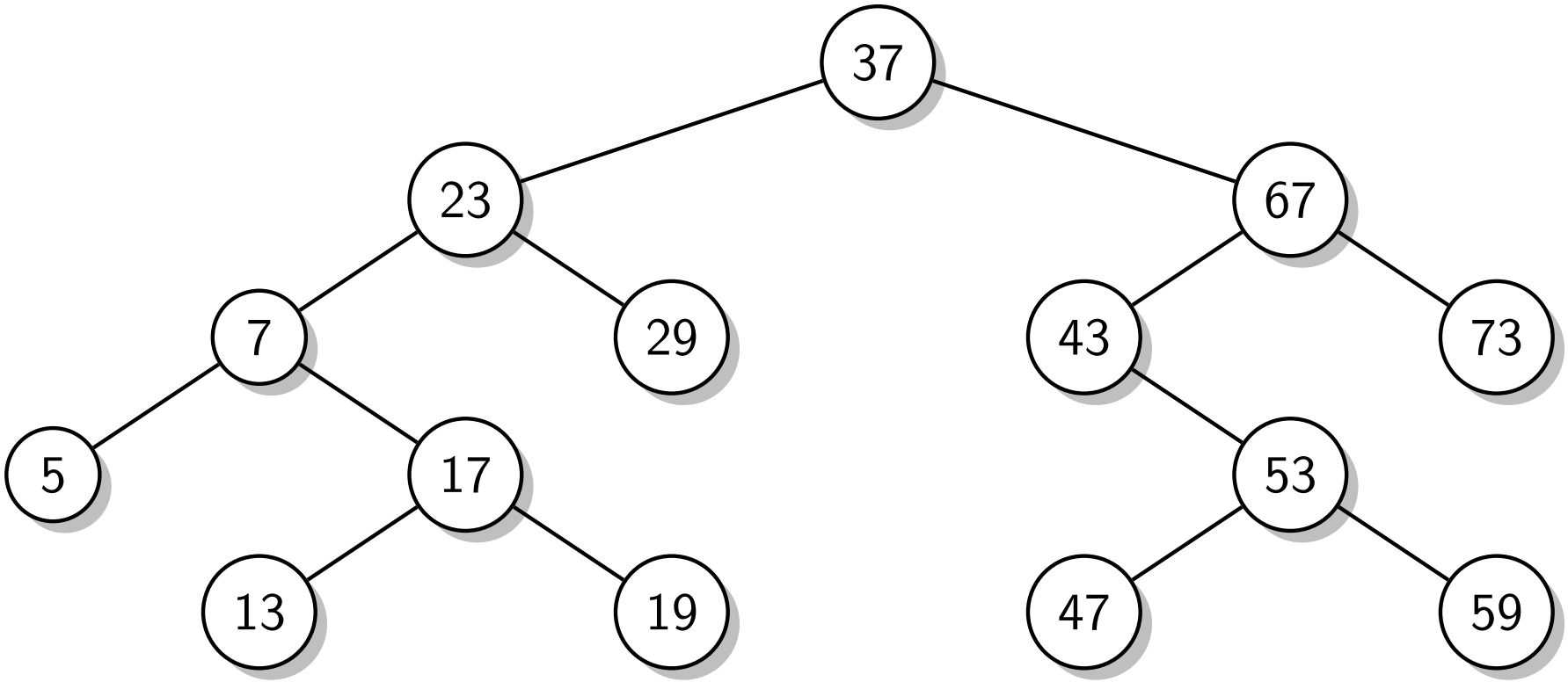
- Navigační pravidlo tedy určuje, jak mají být data v binárním vyhledávacím stromu rozmístěna.
- Znalosti rozmístění dat ve stromu využijeme při jejich vyhledávání.
- Algoritmy pro vložení a vyjmutí ze stromu jsou navázány na algoritmus vyhledávání.
- Binární vyhledávací strom je tedy už od počátku budován s ohledem na toto pravidlo.

Binární vyhledávací strom – vyhledávání

Hledání zahájíme v kořeni stromu r. Potom mohou nastat tyto možnosti:

1. Strom s kořenem r je prázdný ($r = NULL$), potom tento strom nemůže obsahovat uzel s klíčem k a hledání končí neúspěchem.
2. V opačném případě srovnáme klíč k s klíčem kořene právě zkoumaného stromu resp. jeho podstromu r. V případě, že
   - 2.1 $k = \mathrm{klic}(r)$ strom obsahuje uzel s klíčem k. Hledání končí úspěšně;
   - 2.2 $k < \mathrm{klic}(r)$ vzhledem k vlastnostem binárních vyhledávacích stromů, jsou všechny uzly s klíči menšími než je klíč r v jeho levém podstromu, pokračujeme rekurzivně v levém podstromu;
   - 2.3 $k > \mathrm{klic}(r)$ na rozdíl od předchozího případu jsou všechny uzly s klíči většími než je klíč r v pravém podstromu, pokračujeme pravým podstromem.

Binární vyhledávací strom



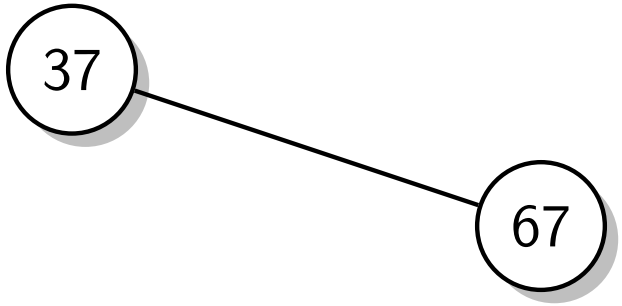
Binární vyhledávací strom – vkládání

- Vložení uzlu (klíče) musí korespondovat s vyhledávacím algoritmem.
- Nejdříve se musíme pokusit vkládaný klíč ve stromu najít.
- Pokud jej nenajdeme, tak místo, kde jsme hledání neúspěšně zakončili odpovídá místu ve stromu, kde by tento klíč měl být.
- Toto plyne z jednoznačnosti cesty mezi kořenem a kterýmkoliv uzlem.
- Nový uzel (klíč) je připojen jako nový list ke stromu – strom roste prostřednictvím listů.
- Otázkou je co s duplicitami? Řešení závisí na povaze konkrétní řešené úlohy.

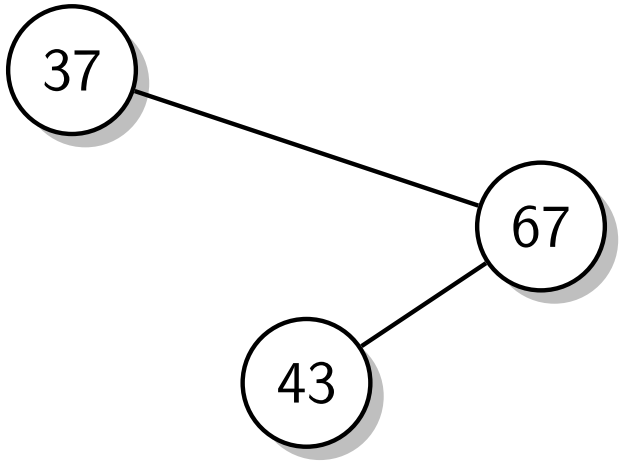
Binární vyhledávací strom – vložení klíče 37



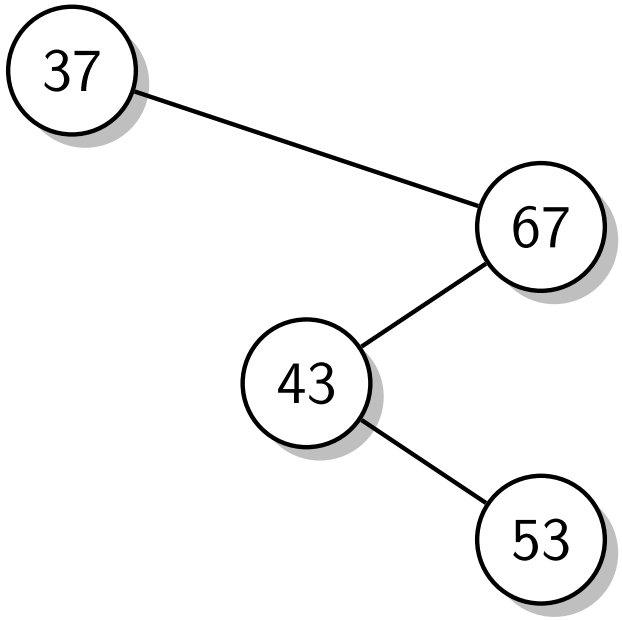
Binární vyhledávací strom – vložení klíče 67



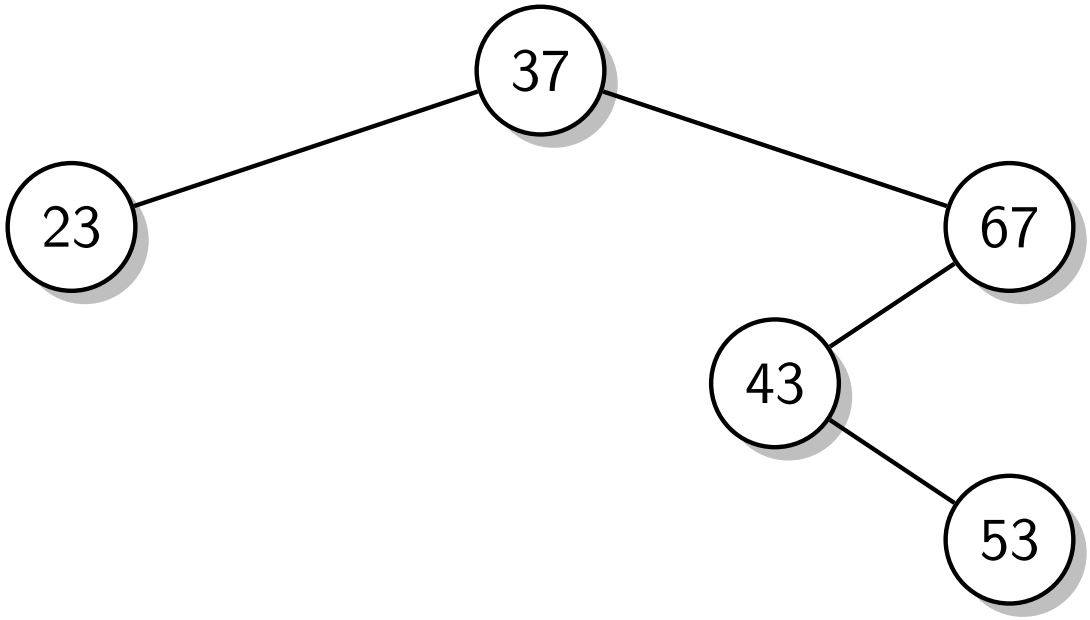
Binární vyhledávací strom – vložení klíče 43



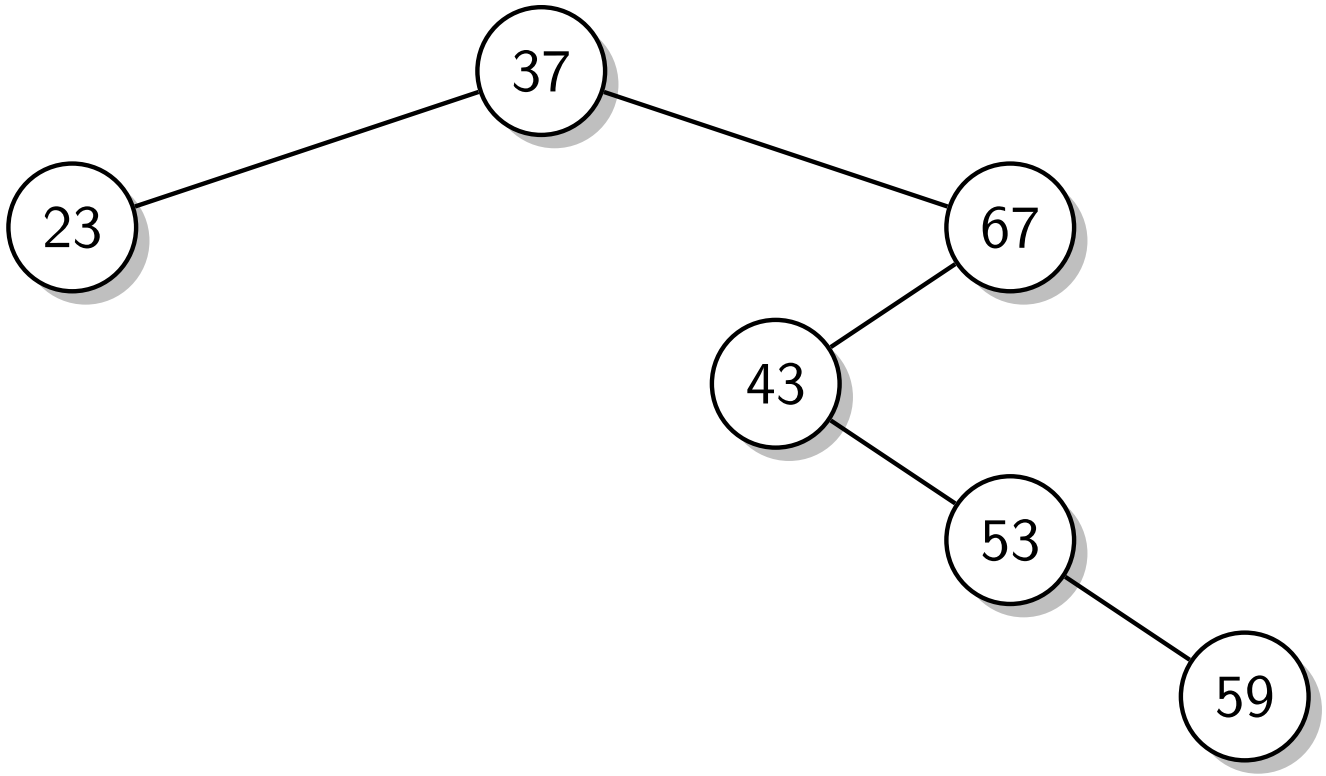
Binární vyhledávací strom – vložení klíče 53



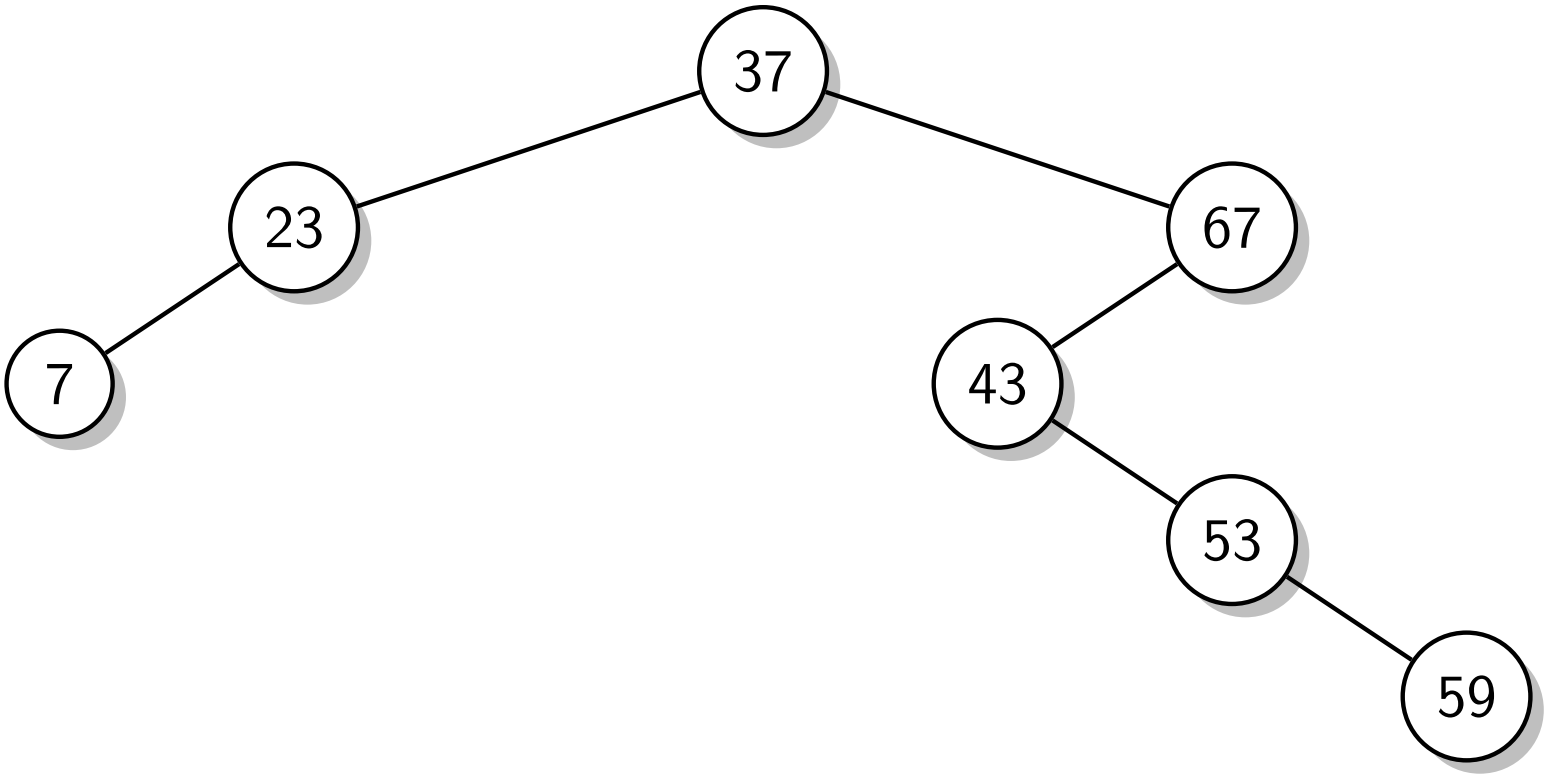
Binární vyhledávací strom – vložení klíče 23



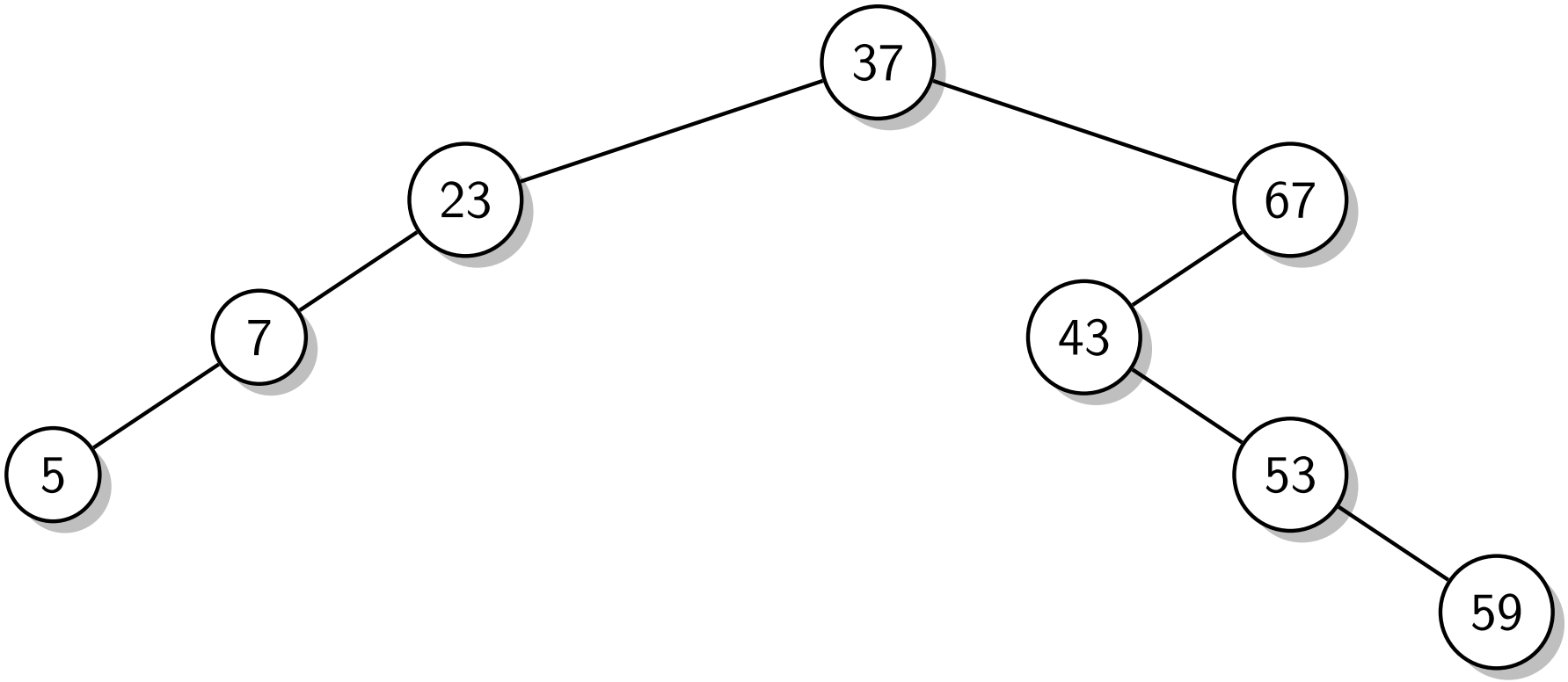
Binární vyhledávací strom – vložení klíče 59



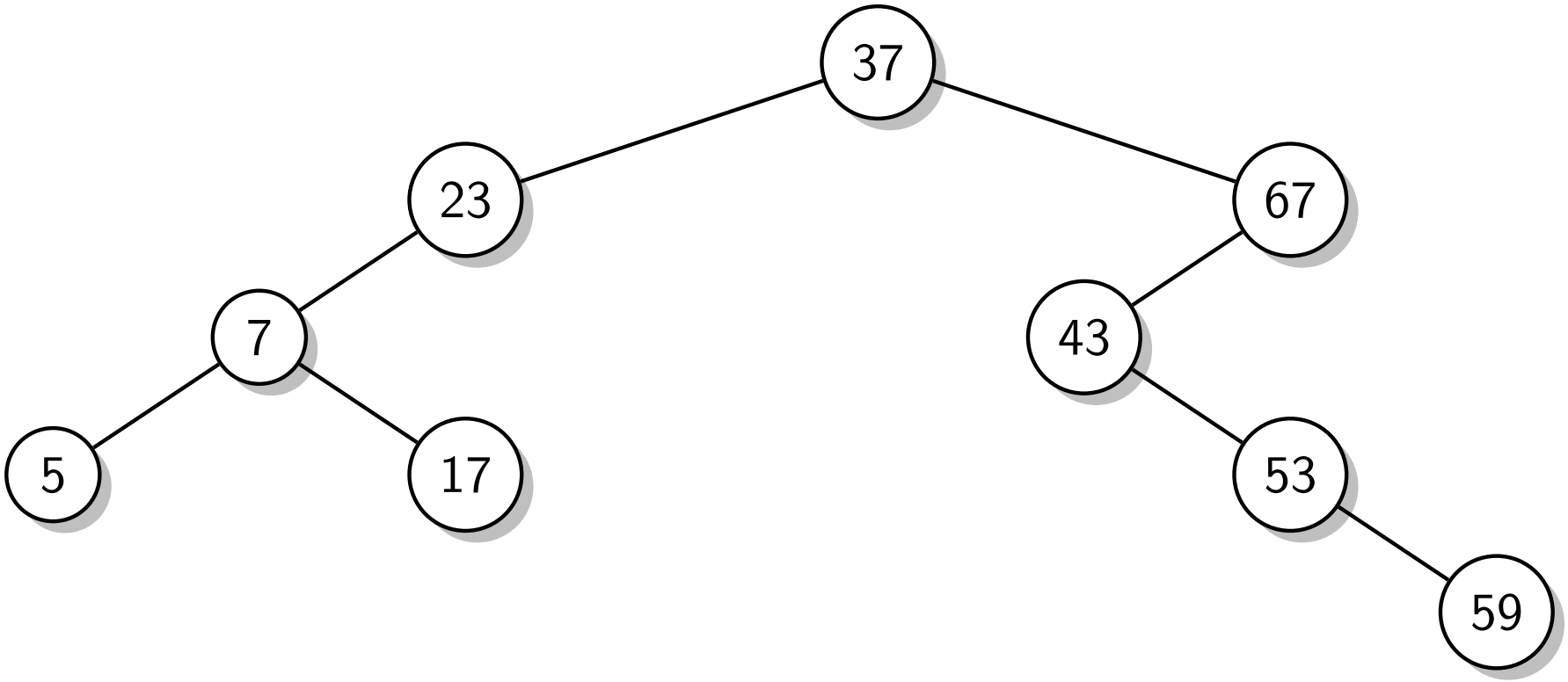
Binární vyhledávací strom – vložení klíče 7



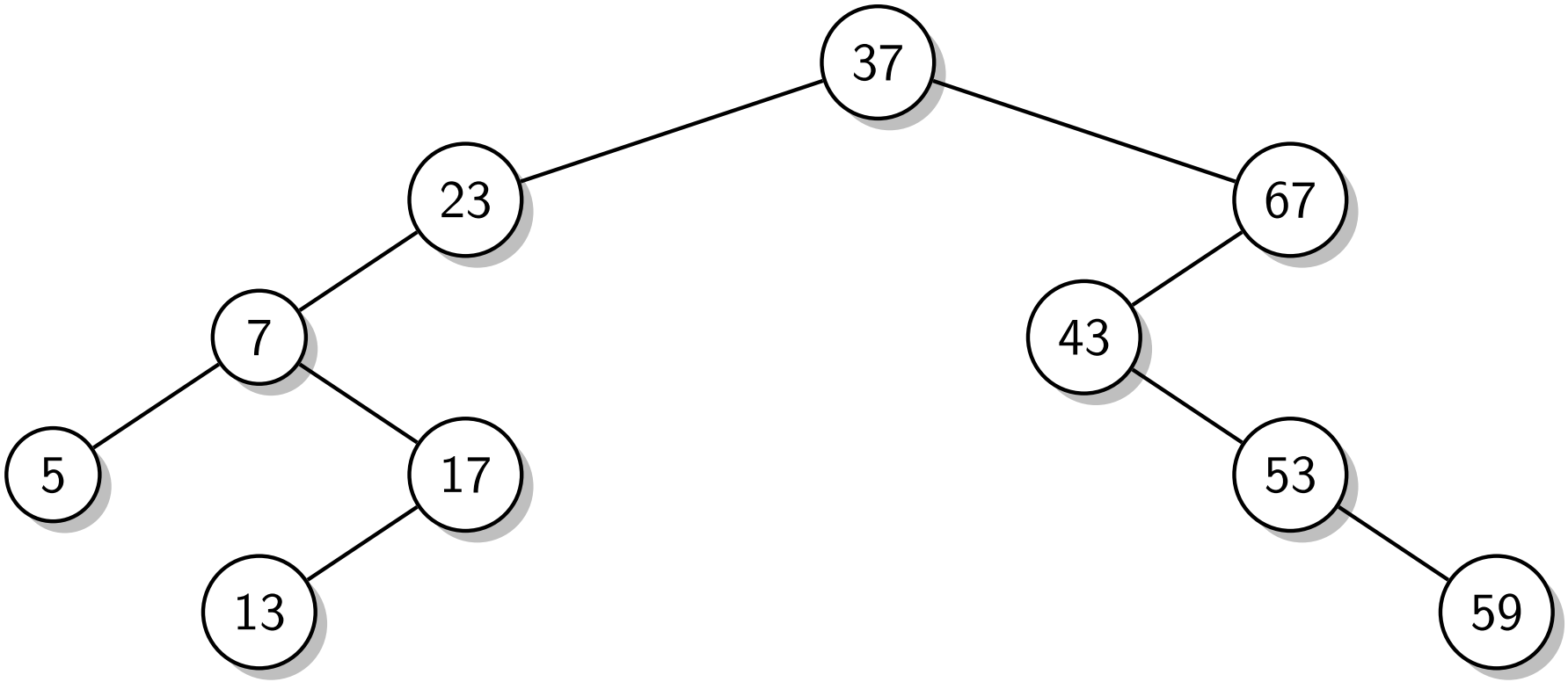
Binární vyhledávací strom – vložení klíče 5



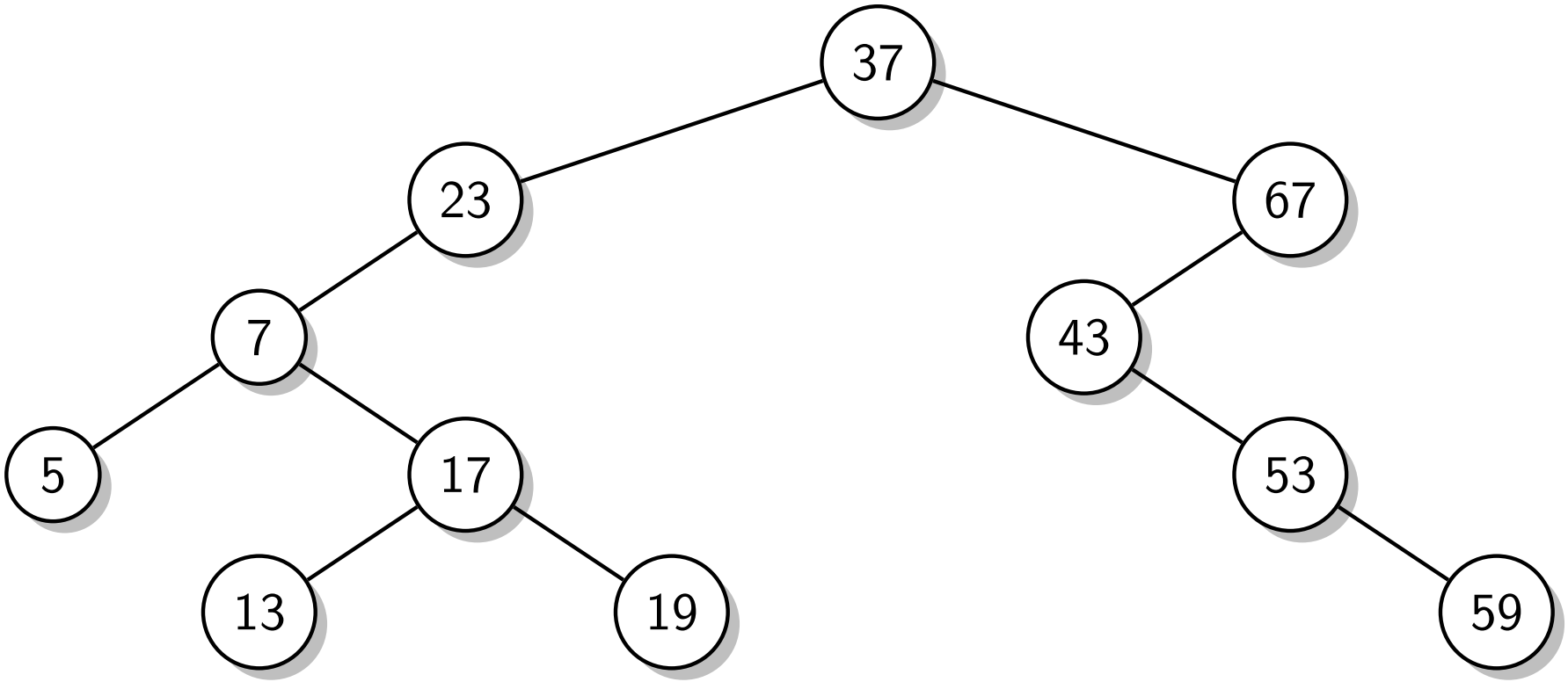
Binární vyhledávací strom – vložení klíče 17



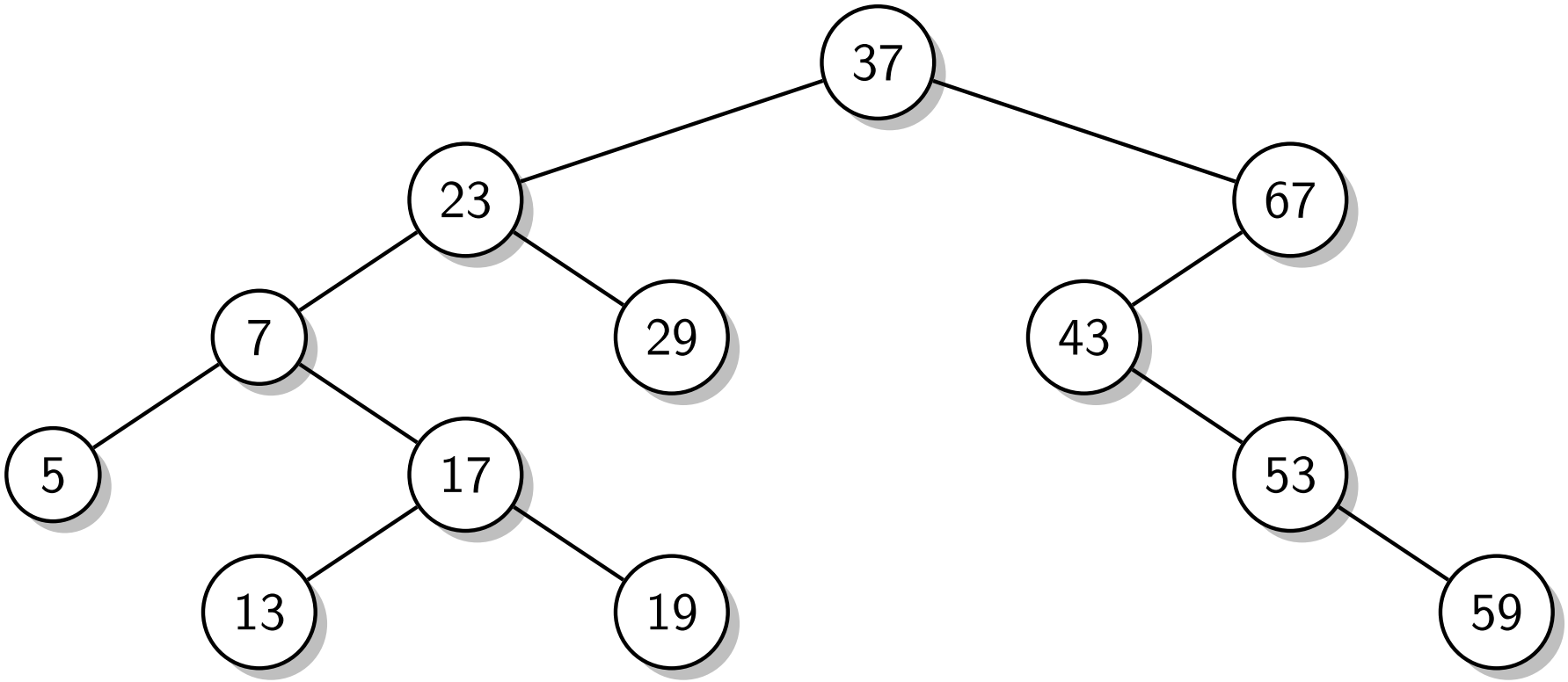
Binární vyhledávací strom – vložení klíče 13



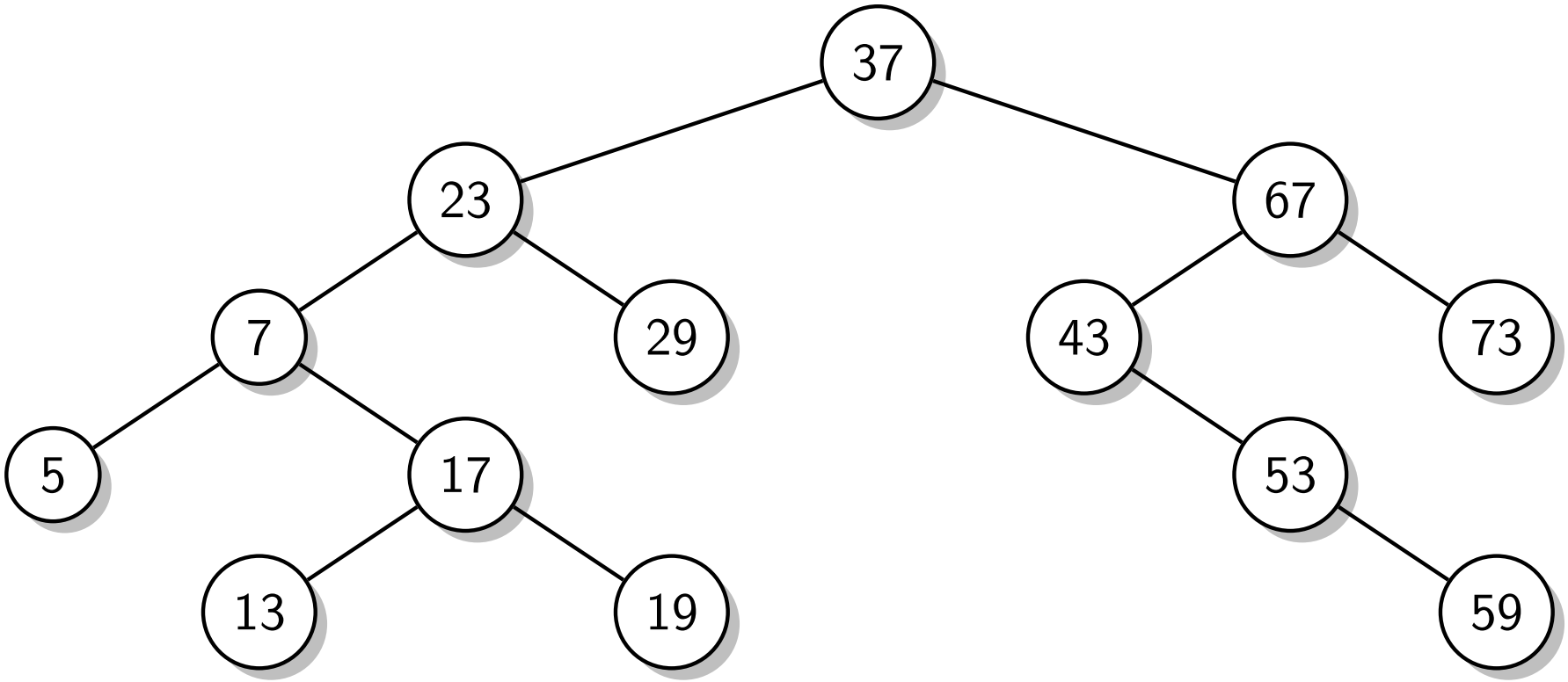
Binární vyhledávací strom – vložení klíče 19



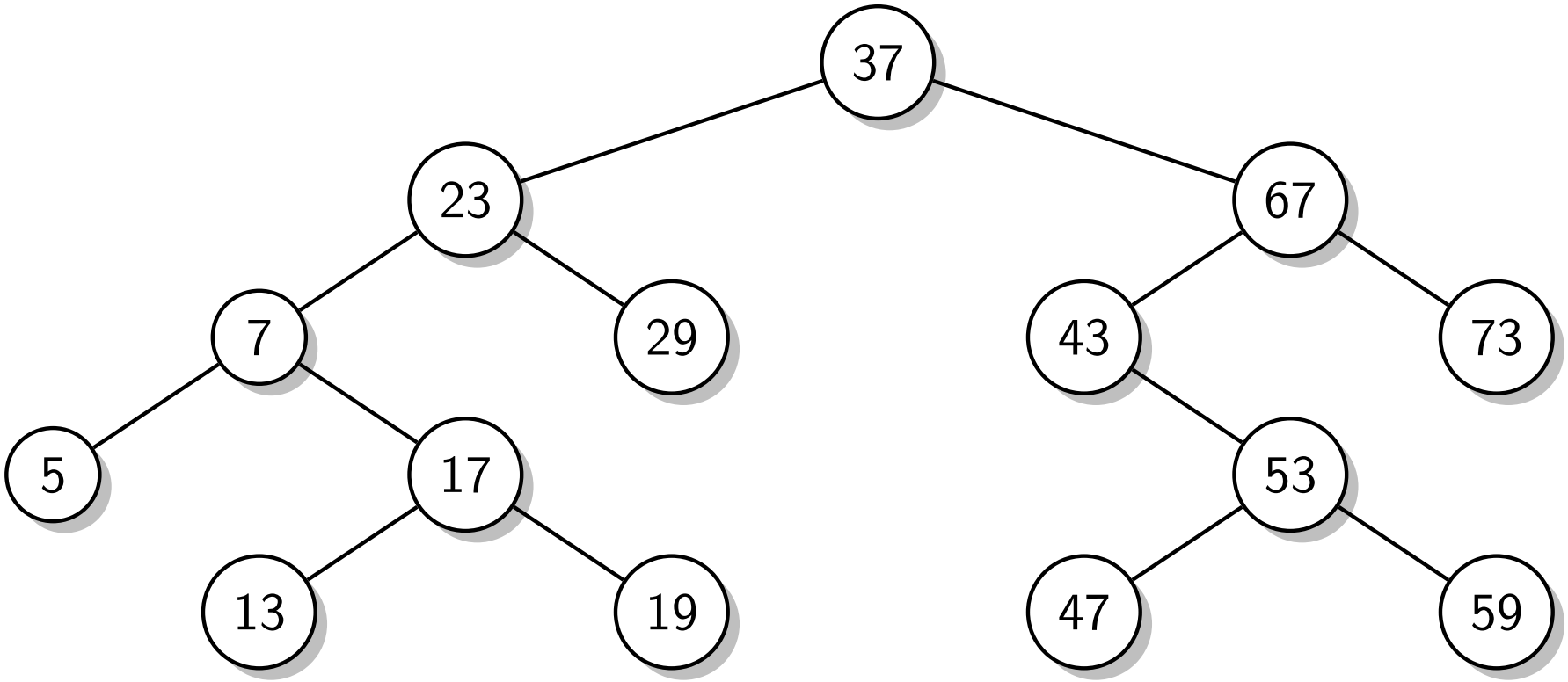
Binární vyhledávací strom – vložení klíče 29



Binární vyhledávací strom – vložení klíče 73



Binární vyhledávací strom – vložení klíče 47



Kontrolní otázky

1. Jak zní rekurzivní definice binárního stromu?
2. Je možné levého a pravého potomka daného uzlu v binárním stromu zaměňovat?
3. Co je to úplný binární strom?
4. Předpokládejme, že máme dán binární strom. Co musí platit pro klíče ve všech uzlech binárního stromu, abychom jej mohli považovat za binární **vyhledávací** strom?
5. Popište rekurzivní algoritmus vyhledání klíče x v binárním vyhledávacím stromu.
6. Z jaké vlastnosti (obecně) stromu plyne fakt, že jsme schopni v binárním vyhledávacím stromu nalézt všechny klíče, které do něj byly vloženy? Jinak řečeno, že lze navštívit všechny uzly ve stromu.

Kontrolní otázky (pokrač.)

7. Z jaké vlastnosti (obecně) stromu plyne fakt, že si nemusíme značkovat jíž navštívené uzly stromu? Například u průchodu grafem jsem si museli uzly označovat.
8. Popište nerekurzivní variantu vyhledání klíče x v binárním vyhledávacím stromu.
9. Popište algoritmus jak lze v binárním vyhledávacím stromu najít minimum mezi všemi v něm uloženými klíči.
10. Popište algoritmus jak lze v binárním vyhledávacím stromu najít maximum mezi všemi v něm uloženými klíči.
11. Popište algoritmus vložení klíče x do binárního vyhledávacího stromu.
12. Jakým způsobem souvisí algoritmus pro vkládání a vyhledávání v binárním vyhledávacím stromu?
13. Jaká je nejhorší podoba binárního vyhledávacího stromu? V jakou jinou datovou strukturu strom v tomto případě zdegeneruje?

Kontrolní otázky (pokrač.)

14. Jak vypadá posloupnost klíčů, jejímž vložením vznikne degenerovaný binární vyhledávací strom?
15. Lze se nějak bránit vzniku degenerovaného binárního vyhledávacího stromu?
16. Máme dáno n různých klíčů. Vložením těchto klíčů do binárního vyhledávacího stromu vznikne degenerovaný binární vyhledávací strom. Jaká je výška takto vzniklého degenerovaného binárního vyhledávacího stromu?
17. Máme dáno n různých klíčů. Tyto klíče vložíme do binárního vyhledávacího stromu, který není degenerovaný. Jaká je, v nejlepším případě, výška takto vzniklého binárního vyhledávacího stromu?
18. Předpokládejme, že máme binární vyhledávací strom s n uzly, který není degenerovaný, a hledáme klíč k. Kolik uzlů musíme v průměru projít, abychom zjistili, zda klíč k v binárním vyhledávacím stromu je nebo není?

Kontrolní otázky (pokrač.)

19. Předpokládejme, že máme binární vyhledávací strom s n uzly, který není degenerovaný, a vkládáme klíč k. Kolik uzlů musíme v průměru projít, než můžeme klíč k do stromu vložit?
20. Popište rekurzivní algoritmus, který zjistí počet uzlů ve stromu. Nápověda: zamyslete se, kolik mají uzlů stromy zmiňované v bodech 1 a 2 v definici binárního stromu.

Děkuji za pozornost