Kopce, kolejka priorytetowa

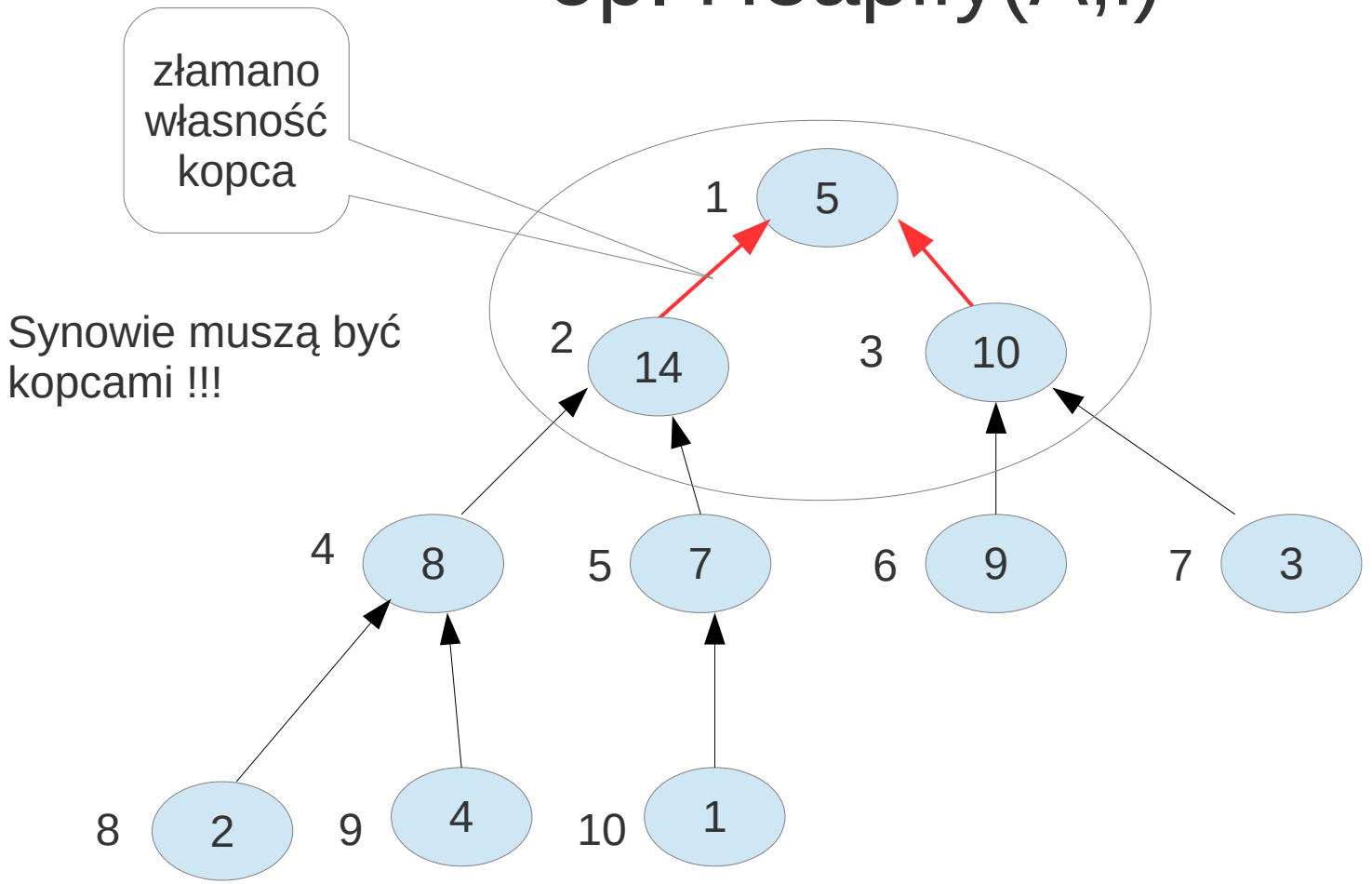
Kopce binarne są przechowywane w zwykłej tablicy „A”...
- „własność kopca”
- podstawowe op. na kopcu: Heapify(A,i), BuildHeap(A)
  - uwaga: nieoczywisty czas działania BuildHeap !!

Patrz opis kopców binarnych

Zastosowania kopców binarnych:
- **kolejka priorytetowa;**
  - elementy oprócz wartości mają „priorytet” (klucz)
  - Insert(S,e), Maximum(S), ExtractMax(S), ChangeKey(S,i,n)
- **sortowanie** w czasie O(n logn), **HeapSort(A)**
  - to jest sortowanie w miejscu!

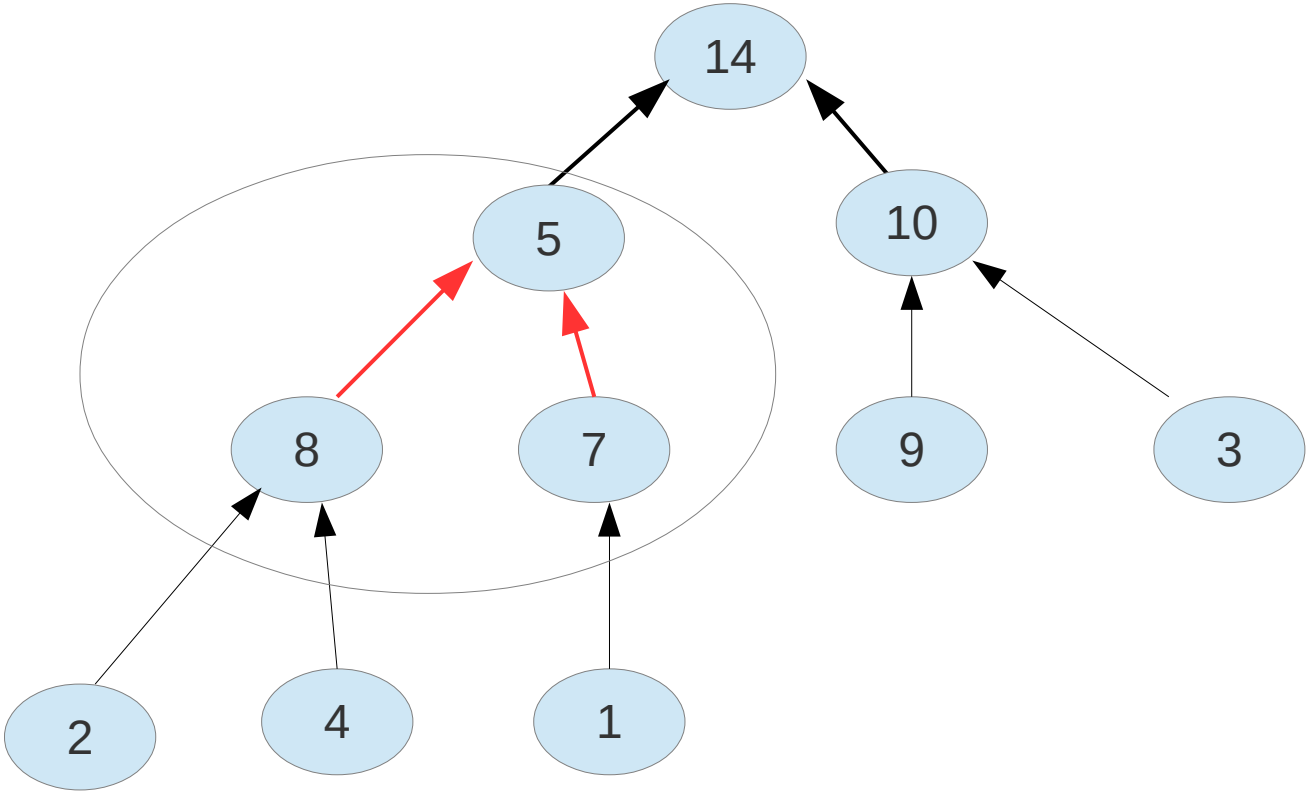
Inne implementacje kolejki priorytetowej:
- z drzewa BST/RB (porównaj czas operacji!)
- inne typy kopców: dwumianowe, Fib,
  - op. Union(A1, A2) – łączy 2 kopce, jest szybsza w tych kopcach!

op. Heapify(A,i)

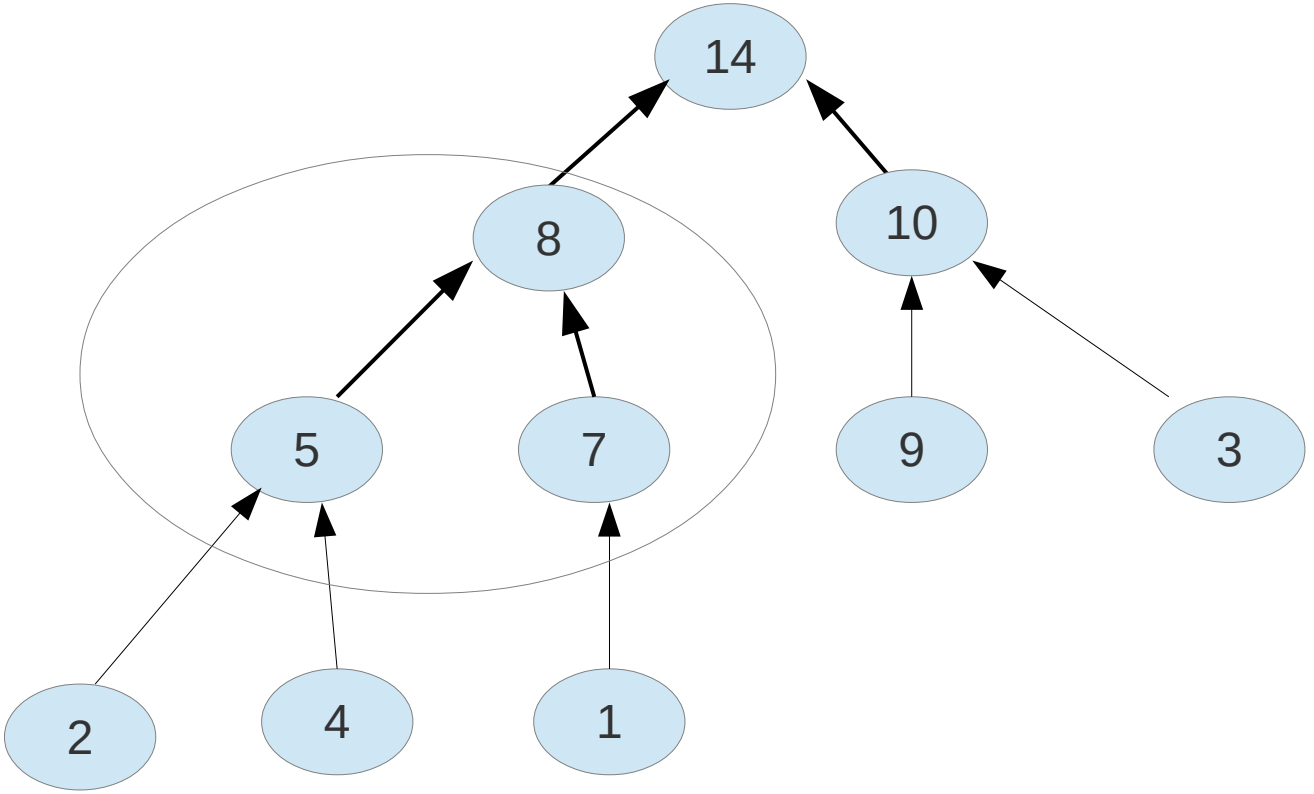


Jak obliczyć indeks lewego/prawego syna dla wierz „i” ?
Jak obliczyć indeks parenta wierz „i” ??

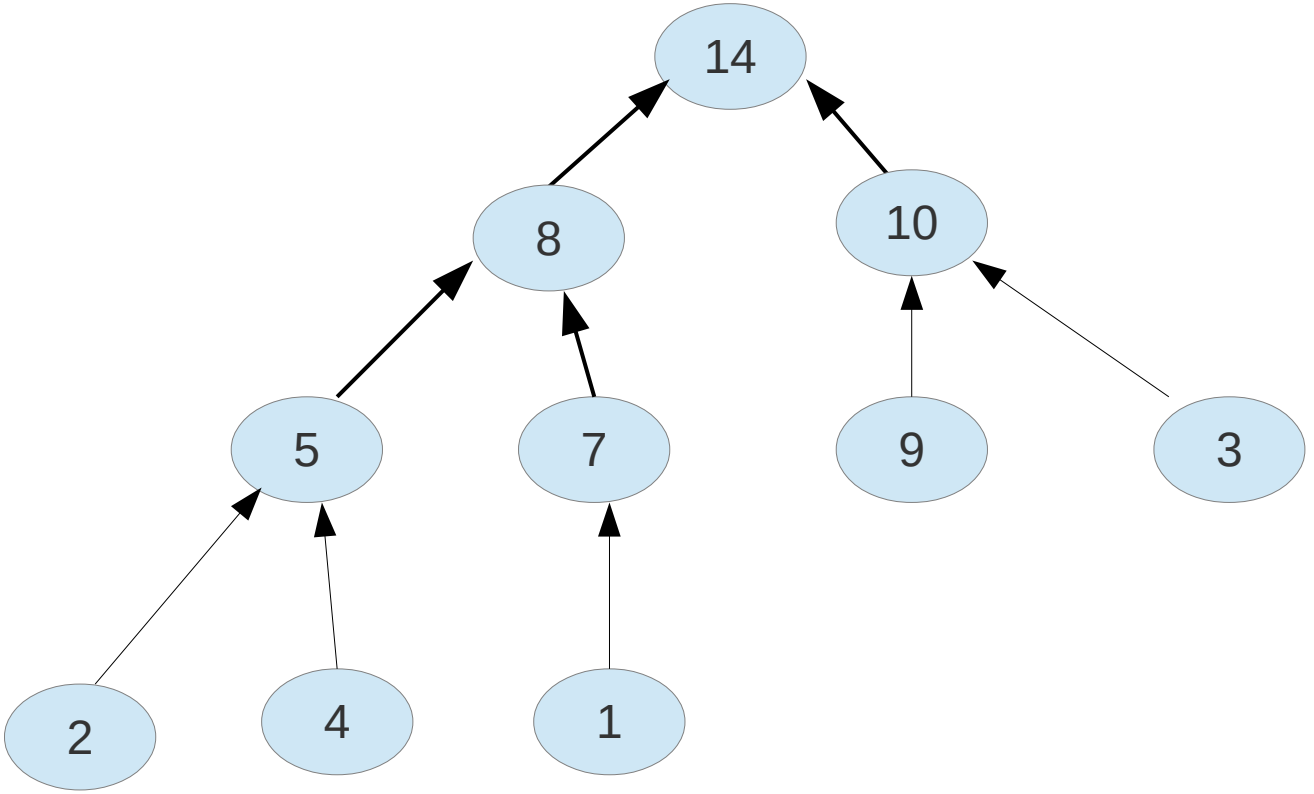
op. Heapify(A,i)



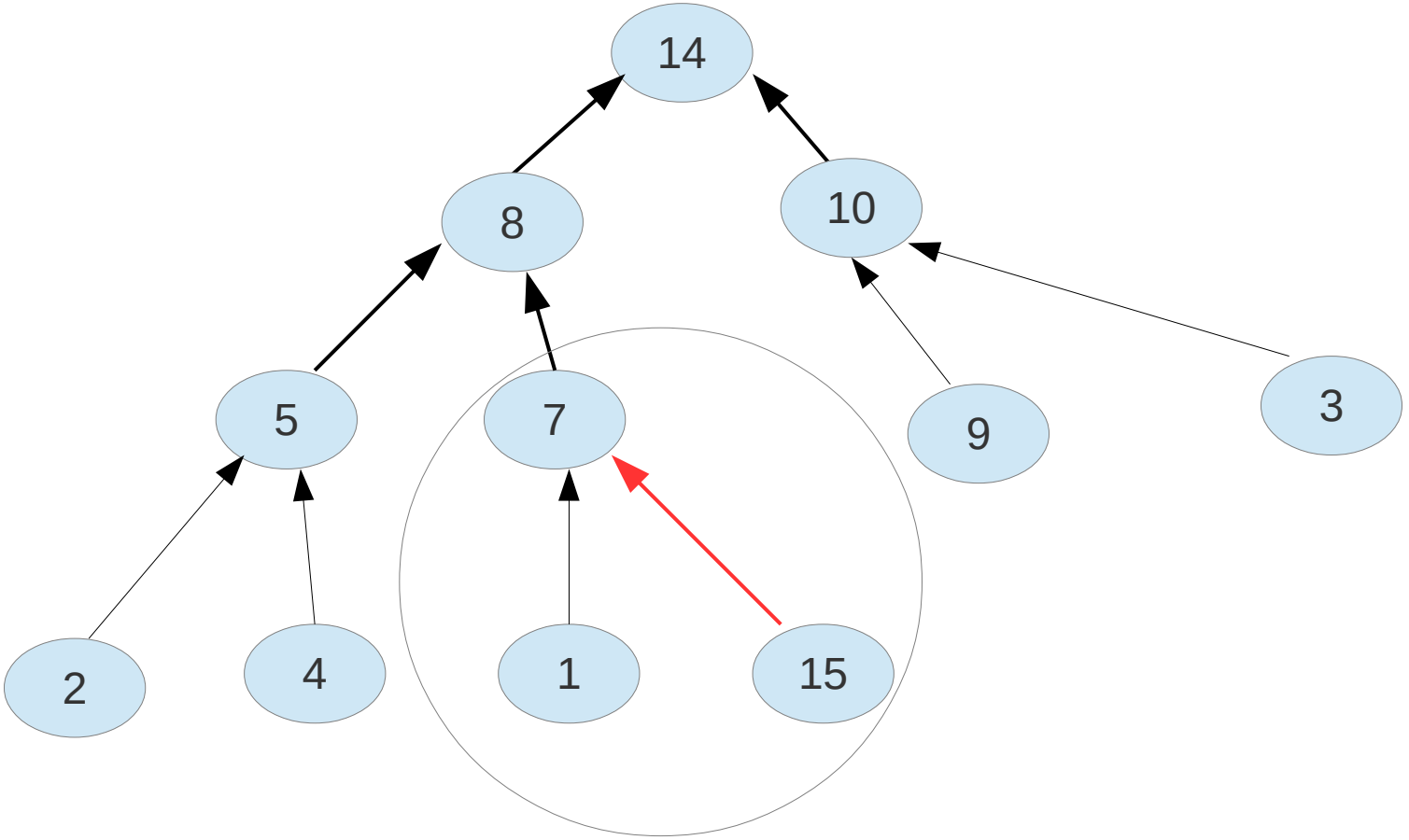
op. Heapify(A,i)



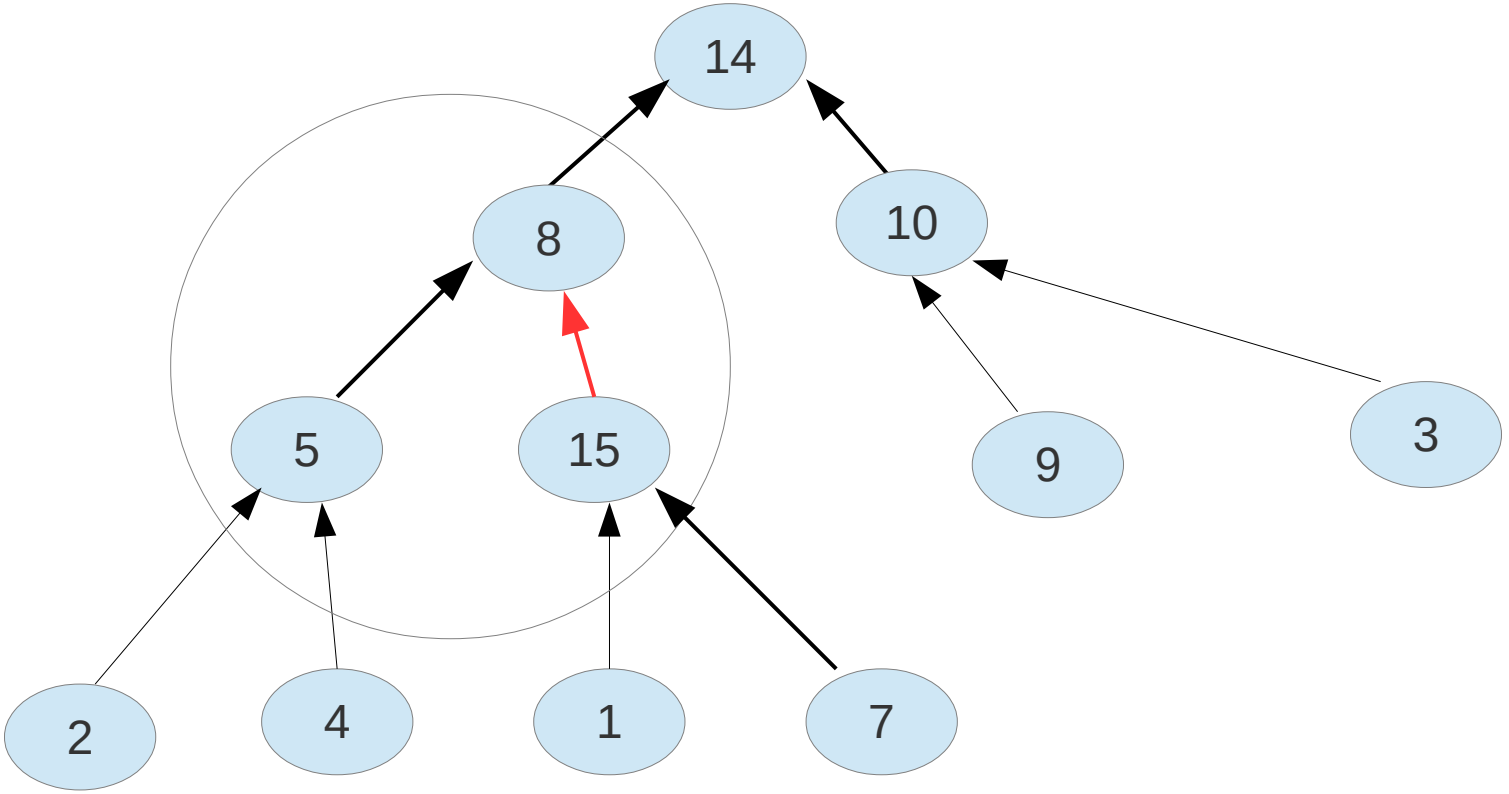
op. HeapInsert(A,15)



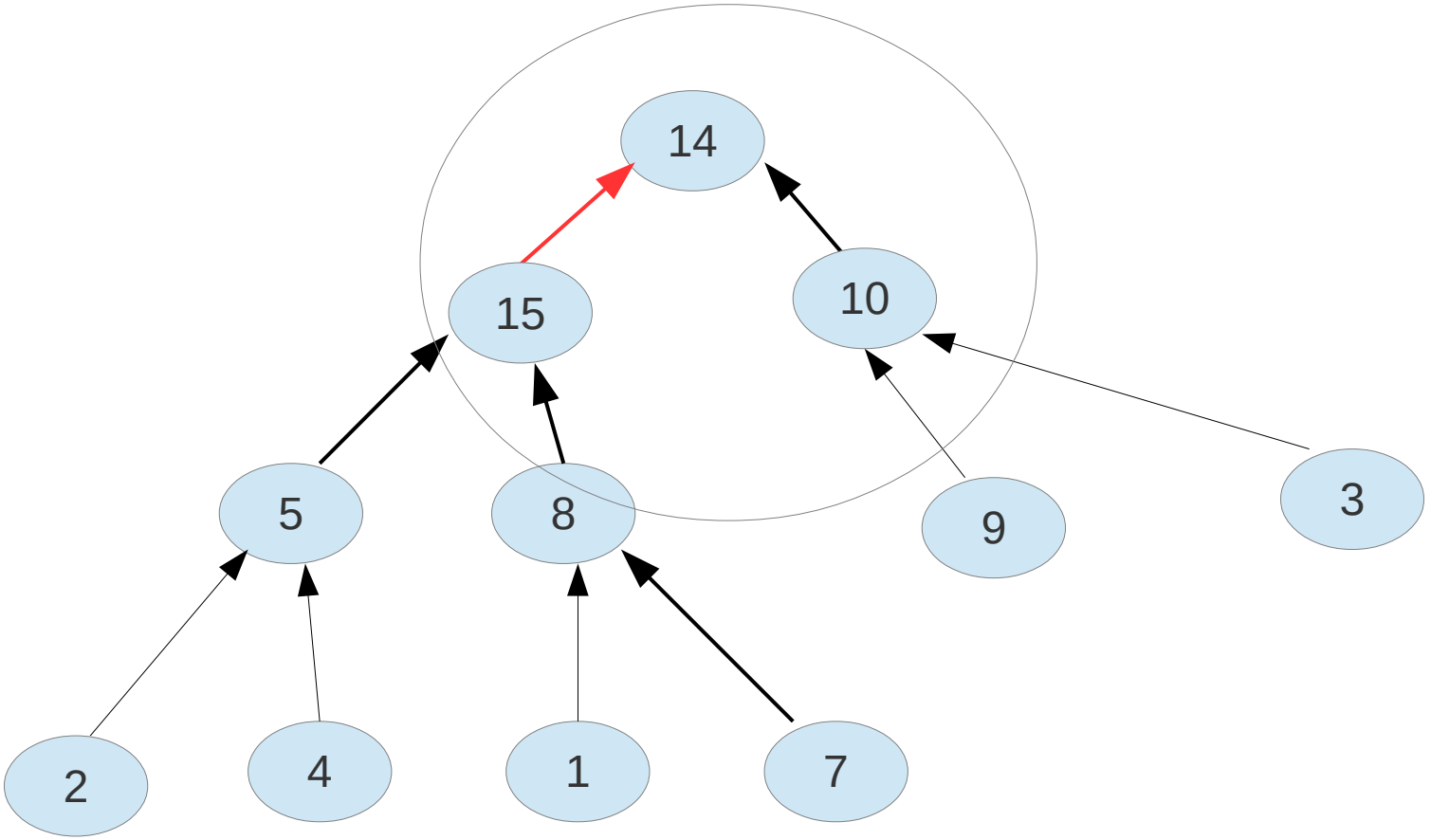
op. HeapInsert(A,15)



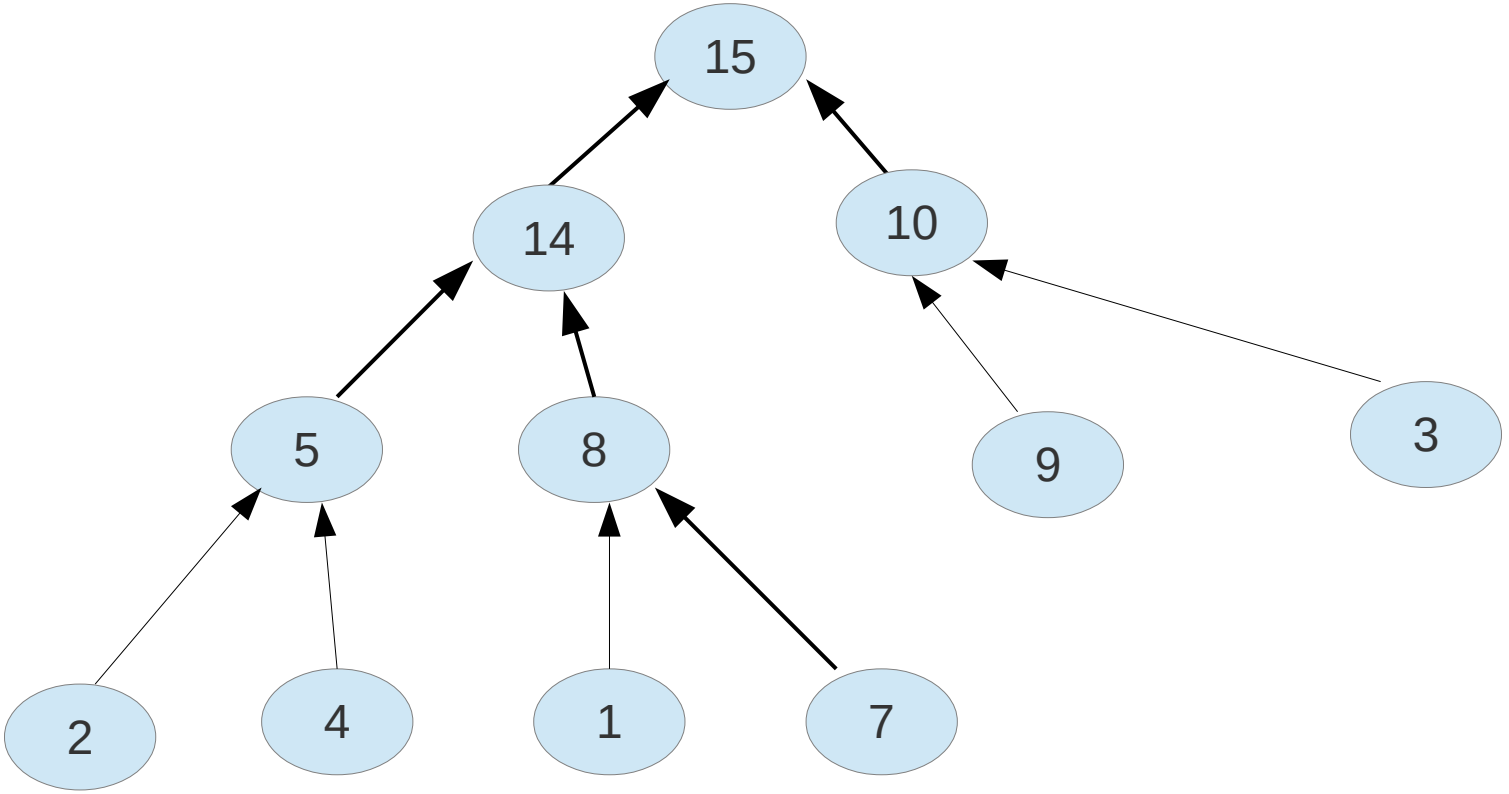
op. HeapInsert(A,15)



op. HeapInsert(A,15)



op. HeapInsert(A,15)



Inne implementacje słownika

Sposób często występujący w praktyce:

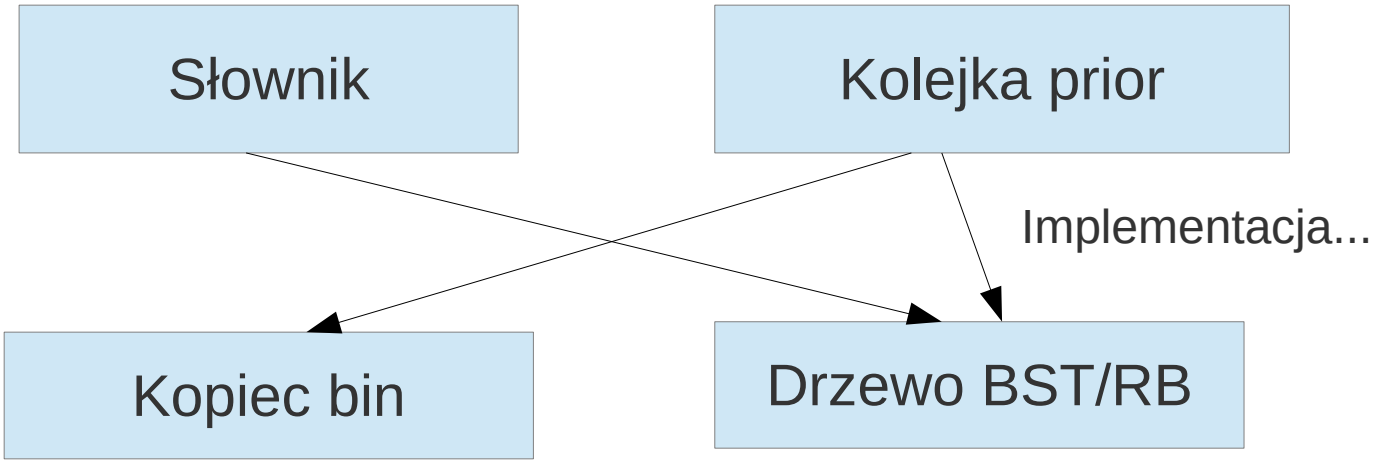
użycie *funkcji haszujących*...

nie ma tu równie silnych gwarancji jak w BST/RB !!!

patrz opis tablic z haszowaniem

oraz adresowania otwartego

słowniki, kolejki prior
vs
kopce bin, drzewa BST/RB



Która implementacja kolejki prior jest lepsza ???

	Kopiec bin	Drzewo BST/RB
ExtractMax	O(logn)	O(logn)
Maximum	O(1)	O(logn)
Insert	O(logn)	O(logn)
Union	O(n)	O(n*logn)
ChangeKey	O(logn)	O(logn)