Skrypt z Algorytmów i struktur danych

Zbiór mniej lub bardziej ciekawych algorytmów i struktur danych, jakie bywały omawiane na wykładzie (albo nie).

praca zbiorowa pod redakcją
Krzysztofa Piecucha

Korzystać na własną odpowiedzialność.

Rozdział 1

To trzeba będzie mądrze podzielić na rozdziały, na razie nie wiem jak

1.1 Algorytm rosyjskich wieśniaków

Todo, todo, todo...

Algorithm 1: Algorytm rosyjskich wieśniaków

Input: a, b - liczby naturalne
Output: $wynik = a \cdot b$

$ap \leftarrow a$
$bp \leftarrow b$
$wynik \leftarrow 0$
while $ap > 0$ **do**
  if $ap \bmod 2 = 1$ **then**
    $wynik \leftarrow wynik + bp$
  end
  $ap \leftarrow ap/2$
  $bp \leftarrow bp \cdot 2$
end

1.2 Algorytm macierzowy wyznaczania liczb Fibonacciego

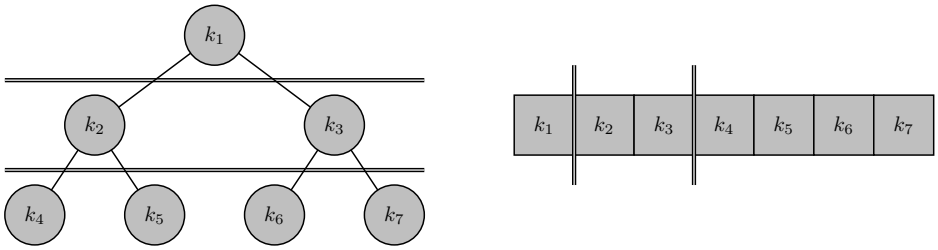
Todo, todo, todo...

Rozdział 2

Kopce binarne

Kopiec binarny to struktura danych, która reprezentowana jest jako prawie pełne drzewo binarne[1] i na której zachowana jest własność kopca. Kopiec przechowuje klucze, które tworzą ciąg uporządkowany. W przypadku kopca typu *min* ścieżka prowadząca od dowolnego liścia do korzenia tworzy ciąg malejący.

Kopce można w prosty sposób reprezentować w tablicy jednowymiarowej –kolejne poziomy drzewa zapisywane są po sobie.

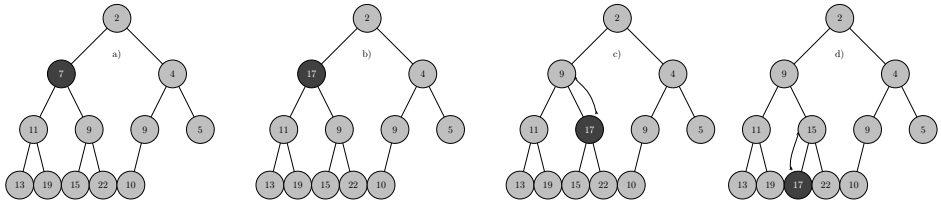


Rysunek 2.1: Reprezentacja kolejnych warstw kopca w tablicy jednowymiarowej.

Warto zauważyć, że tak reprezentowane drzewo pozwala na łatwy dostęp do powiązanych węzłów. Synami węzła o indeksie i są węzły $2i$ oraz $2i+1$, natomiast jego ojcem jest $\lfloor \frac{i}{2} \rfloor$.

Kopiec powinien udostępniać trzy podstawowe funkcje: `zamien_element`, która podmienia wartość w konkretnym węźle kopca, `przesun_w_gore` oraz `przesun_w_dol`, które zamieniają odpowiednie elementy pilnując przy tym, aby własność kopca została zachowana.

[1] To znaczy wypełniony na wszystkich poziomach (poza, być może, ostatnim).



Rysunek 2.2: Przykład działania funkcji `zamien_element`. a) Oryginalny kopiec. b) Zmiana wartości w wyróżnionym węźle. c) Ponieważ nowa wartość jest większa od wartości swoich dzieci, należy wykonać wywołanie funkcji `przesn_w_dol`. d) Po zmianie własność kopca nie jest zachowana, dlatego należy ponownie wywołać funkcję `przesn_w_dol`. To przywraca kopcowi jego własność.

Algorithm 2: Implementacja funkcji `zamien_element`

```
if k[i] < v then
    k[i] = v;
    przesun_w_dol(k, i);
end
else
    k[i] = v;
    przesun_w_gore(k, i);
end
```

;

Dodatek A

Porównanie programów przedmiotu AiSD na różnych uczelniach

	UWr	UW	UJ	MIT	Oxford
Stosy, kolejki, listy		✓			
Dziel i zwyciężaj	✓				
Programowanie Dynamiczne	✓	✓	✓	✓	
Metoda Zachłanna	✓	✓	✓		
Koszt zamortyzowany	✓	✓			✓
NP-zupełność	✓	✓		✓	
PRAM / NC	✓				
Sortowanie	✓	✓			
Selekcja	✓	✓			
Słowniki	✓	✓	✓		✓
Kolejki priorytetowe	✓	✓			
Hashowanie	✓	✓			
Zbiory rozłączne	✓				
Algorytmy grafowe	✓	✓	✓	✓	✓
Algorytmy tekstowe	✓	✓			
Geometria obliczeniowa	✓				
FFT	✓				✓
Algorytm Karatsuby	✓			✓	
Metoda Newtona				✓	
Algorytmy randomizowane	✓				✓
Programowanie liniowe					✓
Algorytmy aproksymacyjne	✓				✓
Sieci komparatorów	✓				
Obwody logiczne	✓				