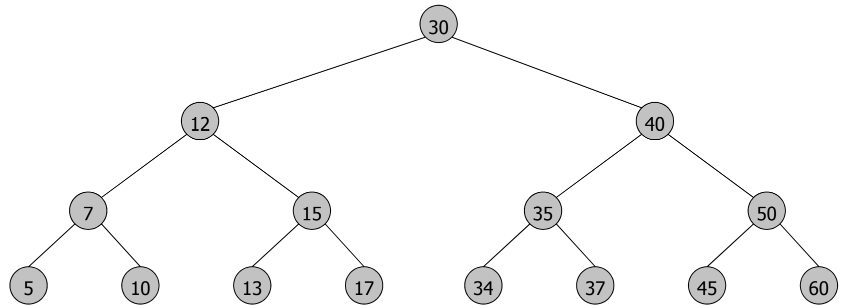
Indeksowanie: B-drzewa

Tadeusz Pankowski
www.put.poznan.pl/~tadeusz.pankowski

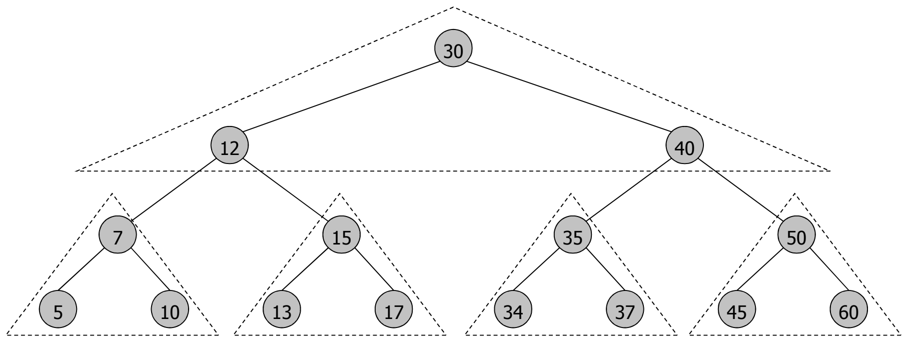
B-drzewa: idea

Drzewo decyzyjne, przeszukiwania binarnego:
F = {5, 7, 10, 12, 13, 15, 17, 30, 34, 35, 37, 40, 45, 50, 60}



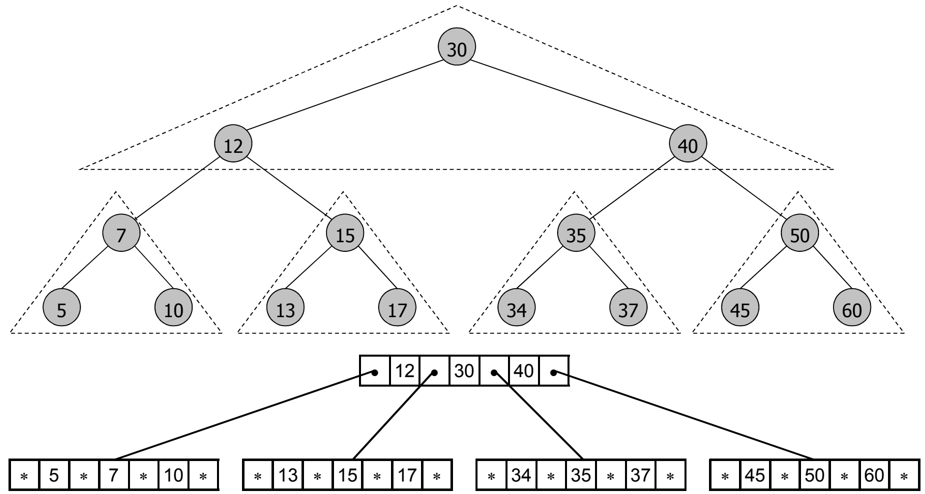
$$\lceil \log_2(N+1) \rceil \le h \le 1 + \lceil \log_2 N \rceil$$

B-drzewa: idea



Sposób grupowania wierzchołków uzasadniający postać B-drzewa

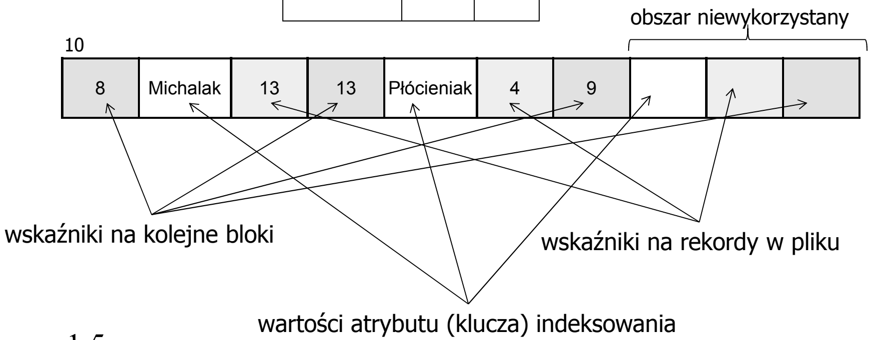
B-drzewa: idea



Drzewo wielodrogowe utworzone w wyniku pogrupowania drzewa w bloki trójelementowe

Postać bloków w indeksie o strukturze B-drzewa

	Nazwisko	RPTR	BPTR
			8
10	Michalak	13	13
	Płócieniak	4	9



$m = 1.5$

W bloku mogą być co najwyżej 3 ($2m$) elementy indeksu i co najmniej 1 ($\lfloor m \rfloor$).
Dalej będziemy zakładać, że m jest **liczbą całkowitą**.

B-drzewo (*perfectly balanced multiway tree*)

Definicja

Niech $h \geqslant 0$ i $m \geqslant 1$. Drzewo (skierowane) T nazywamy B-drzewem klasy $t(h, m)$, co zapisujemy $T \in t(h, m)$, jeżeli T jest drzewem pustym ($h = 0$) lub spełnione są następujące warunki:

1. Wszystkie drogi prowadzące z korzenia drzewa do liści są jednakowej długości równej h, liczbę h nazywamy *wysokością* drzewa T.
2. Każdy wierzchołek, z wyjątkiem korzenia i liści, ma co najmniej $m + 1$ synów – korzeń jest albo liściem, albo ma co najmniej dwóch synów.
3. Każdy wierzchołek ma co najwyżej $2m + 1$ synów.

B-drzewa

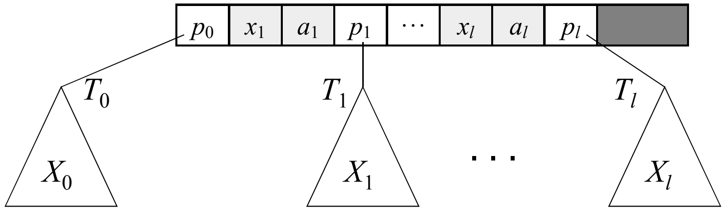
1. B-drzewo (*perfectly balanced multiway tree*) jest drzewem wyszukiwania o takich ograniczeniach, które zapewniają, że jest ono zawsze **w pełni zrównoważone**, a stopień jego wypełnienia nie jest nigdy zbyt mały. (Komplikuje to oczywiście algorytmy wstawiania i usuwania, ale przyspiesza wyszukiwanie):
   - wszystkie drogi od korzenia do liści są jednakowej długości **h**,
   - każdy wierzchołek z wyjątkiem korzenia ma, co najmniej **m+1** następników (poddrzew),
   - każdy wierzchołek ma, co **najwyżej 2m+1** następników (poddrzew),
   - korzeń ma, co najmniej 2 następniki poddrzewa.
2. Umożliwiają wyszukiwanie, dołączanie i usuwanie w czasie logarytmicznym, $O(\log_m N)$
3. B-drzewa i ich warianty (B+-drzewa) stanowią podstawę implementacji indeksów w komercyjnych systemach baz danych.

B-drzewa – struktura wierzchołków

Struktura wierzchołka (bloku, strony) w B-drzewie. Wszystkie bloki mają jednakową wielkość.

p_0	x_1	a_1	p_1	$\cdots$	x_l	a_l	p_l	

1. l – liczba elementów indeksu w bloku, tj. liczba par (x_i, a_i), przy czym
   - $1 \leqslant l \leqslant 2m$, dla korzenia,
   - $m \leqslant l \leqslant 2m$, dla wierzchołków różnych od korzenia.
2. x_i – wartość klucza (atrybutu) indeksowania. Wartości klucza zawarte w bloku są uporządkowane:
$$x_1 \leqslant x_2 \leqslant \cdots \leqslant x_l.$$
3. a_i – wskaźnik na rekord, dla którego klucz indeksowania ma wartość a_i.
4. p_i – wskaźnik na blok będący korzeniem poddrzewa T_i.



Niech X_i oznacza zbiór wszystkich wartości klucza indeksowania zawarte w poddrzewie T_i wskazywanym przez wskaźnik p_i. Wówczas:

- wszystkie wartości w X_0 są nie większe niż x_1,

$$\forall x \in X_0 \; (x \leqslant x_1),$$

- x_i jest nie mniejszy od wartości w X_{i-1} i nie większy od wartości w X_i,

$$\forall x' \in X_{i-1} \forall x'' \in X_i \; (x' \leqslant x_i \leqslant x'',$$

- x_l jest nie większy od wartości zawartych w X_l,

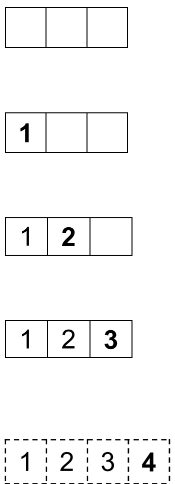
$$\forall x \in X_l \; (x_l \leqslant x).$$

B-drzewa – operacja dołączania

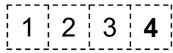
➡ **Dołączanie** • **metoda podziału**
• **metoda kompensacji**

1. Chcemy dołączyć element indeksu o kluczu X, tak aby nie naruszyć struktury B-drzewa.
2. Dołączanie poprzedzone jest procedurą SZUKAJ, w wyniku której, albo znajdziemy wierzchołek zawierający klucz X (koniec) albo znajdziemy adres wierzchołka (**liścia**) do którego należy dołączyć klucz X.
3. Jeśli wierzchołek ma mniej niż 2m elementów to dołączamy nowy klucz.
4. Jeśli wierzchołek ma 2m elementów to następuje kolizja (przepełnienie, nadmiar), którą rozwiązujemy albo metodą **podziału** albo metodą **kompensacji**.

B-drzewo klasy t(1,1.5):



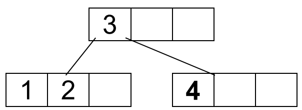
B-drzewo klasy t(h,m) – **metoda podziału** bloku wierzchołka:

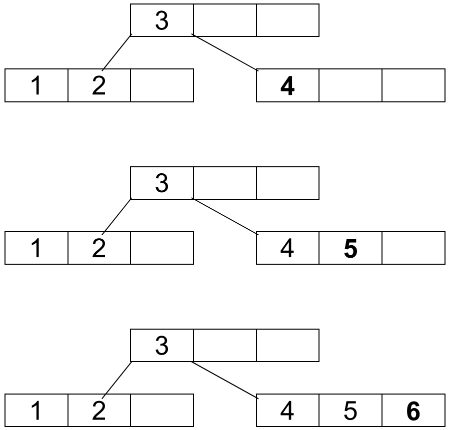


W bloku znajduje się 2m+1 elementów indeksu – nadmiar, przepełnienie.
Postępowanie:

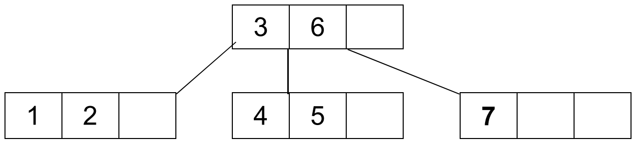
1. Jako element środkowy przyjmujemy element na pozycji k=⌈m+1⌉
2. W starym bloku, b, pozostawiamy elementy o numerach: 1, ..., k – 1.
3. Tworzymy nowy blok, b', i umieszczamy w nim elementy o numerach k+1,...,2m+1.
4. Element środkowy przesuwany jest do bloku poprzednika (rodzica).
5. Dla przesuwanego elementu może powstać konieczność utworzenia nowego korzenia. Wówczas otrzymujemy drzewo klasy t(h+1,m).

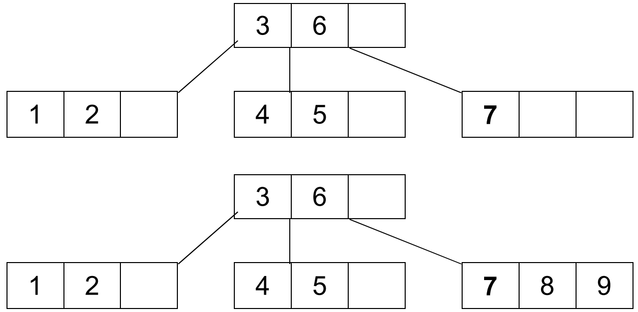
B-drzewo klasy t(2, 1.5):



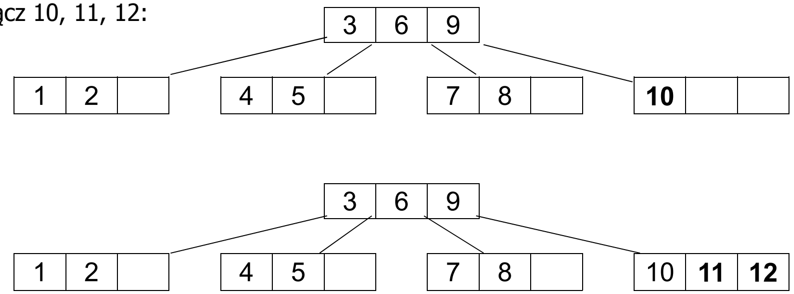


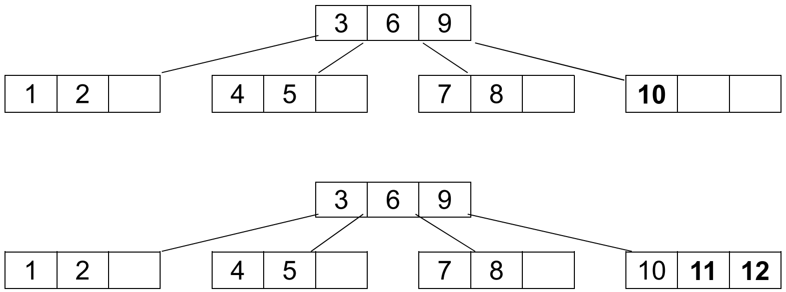
Dołącz 7:



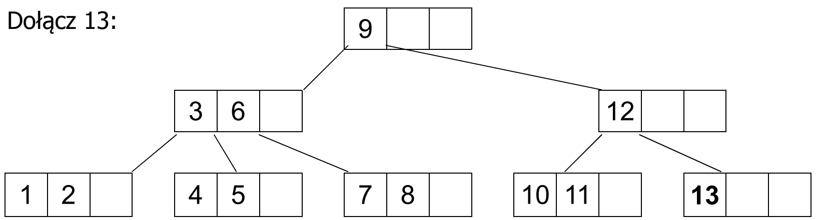


Dołącz 10, 11, 12:

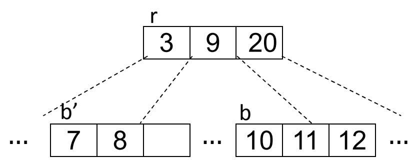




Dołącz 13:

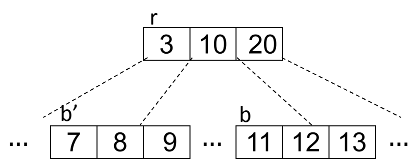


B-drzewo klasy t(h,m) – **metoda kompensacji**:

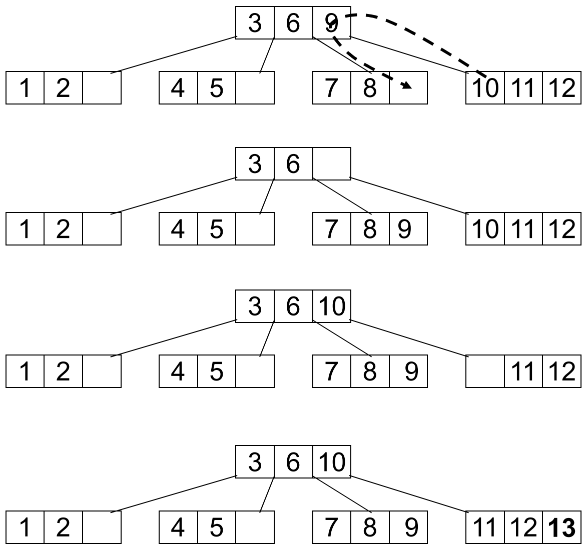


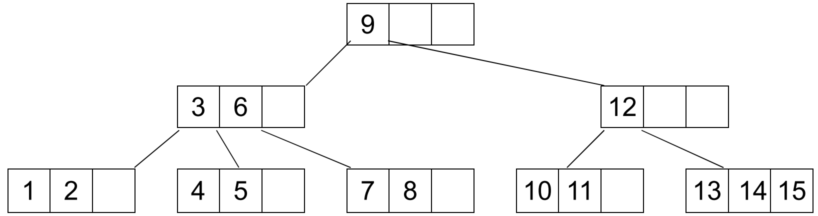
W bloku **b** znajduje się 2m elementów indeksu – próba dołączenia elementu x=13 prowadzi do nadmiaru (przepełnienia).
W bloku **b’** po lewej stronie liczba elementów jest mniejsza niż 2m.
W bloku **r** lewy i prawy wskaźnik elementu x’ (9) wskazują (bezpośrednio lub pośrednio) na bloki **b’** i **b**.
Postępowanie:

1. Element x’ (np. 9) dołączany jest do bloku b’ (na ostatnim miejscu).
2. Do bloku **r** w miejsce elementu x’ wstawiany jest najmniejszy element z bloku **b** (np. 10) i element ten usuwany jest z **b**.
3. Do bloku b można teraz bezkonfliktowo dołączyć x (13).

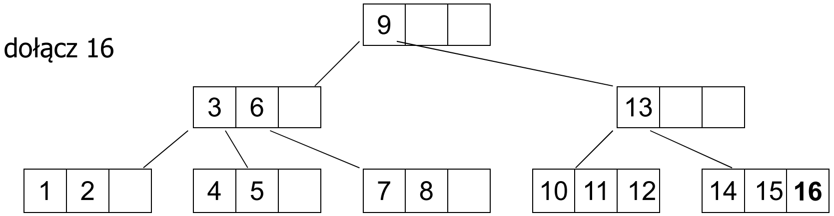


Dołącz 13:

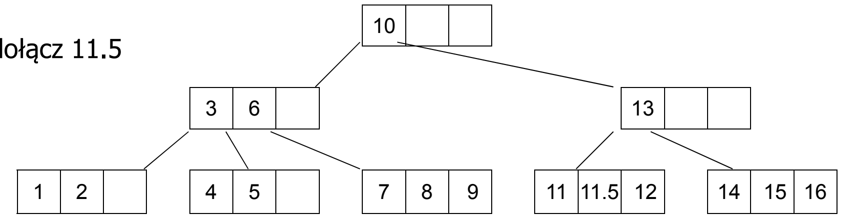




dołącz 16



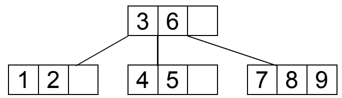
dołącz 11.5



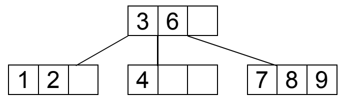
B-drzewa – operacja usuwania

- **Usuwanie** • **metoda łączenia**
  • **metoda kompensacji**

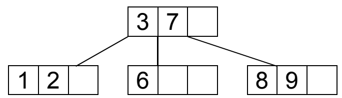
1. Chcemy usunąć element o kluczu **x**. Usuwanie podobnie jak dołączanie poprzedzone jest algorytmem wyszukiwania. Procedura SZUKAJ powinna zakończyć się powodzeniem i zwrócić adres (b) wierzchołka (bloku) zawierającego klucz **x**.
   - Jeśli strona jest liściem to usuwamy element indeksu o kluczu x. Może wówczas wystąpić **niedomiar**, który usuwamy **metodą łączenia lub kompensacji.**
   - Jeżeli strona nie jest liściem to przeglądamy poddrzewo wskazywane przez **prawy** wskaźnik stojący **za** kluczem x i szukamy najmniejszego indeksu (L) - idziemy ścieżką wskazywaną przez P_0, aż dojdziemy do liścia. Ten najmniejszy element wstawiamy w miejsce **x** a następnie usuwamy go z liścia – może wystąpić niedomiar.



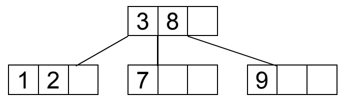
usuń 5:

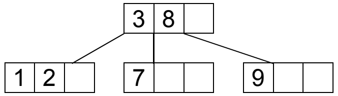


usuń 4: kompensacja (z prawym)

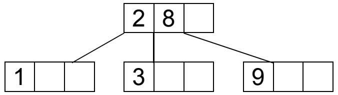


usuń 6: kompensacja (z prawym)

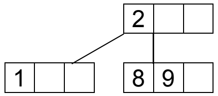




usuń 7: kompensacja (z lewym)



usuń 3: łączenie

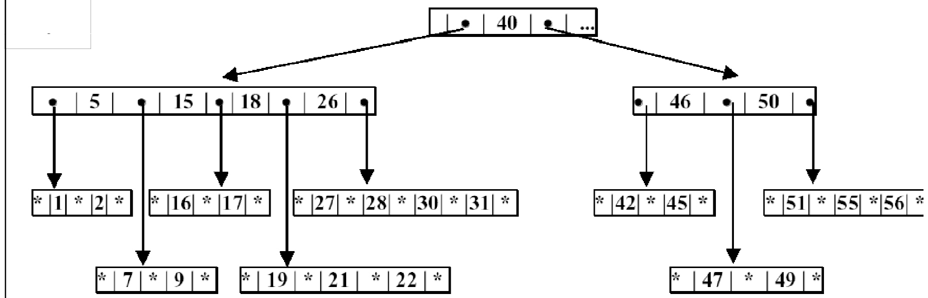


usuń 2: łączenie

B-drzewa – zadanie

Dane jest B-drzewo klasy **t(3, 2)**:

- wstaw obiekt o kluczu **32** (metodą podziału i metodą kompensacji),
- usuń obiekt o kluczu **46** (metodą łączenia i metodą kompensacji).



B-drzewa – złożoność

B-drzewo klasy t(h,m).

Obliczenie liczby wierzchołków (bloków) w B-drzewie:

Poziom i	Liczba wierzchołków przy min wypełnieniu W_{min}	Liczba wierzchołków przy max wypełnieniu W_{max}
1	1	1
2	2	$2m+1$
3	$2(m+1)$	$(2m+1)^2$
...	...	...
h	$2(m+1)^{h-2}$	$(2m+1)^{h-1}$
Σ	$1+\frac{2}{m}((m+1)^{h-1}-1)$	$\frac{1}{2m}((2m+1)^h-1)$

B-drzewa – złożoność (c.d.)

B-drzewo klasy t(h,m)

Obliczenie liczby elementów indeksu w B-drzewie:

- W_{min} – liczba wierzchołków przy minimalnym wypełnieniu
- W_{max} – liczba wierzchołków przy maksymalnym wypełnieniu
- N_{min} – liczba elementów indeksu przy minimalnym wypełnieniu
- N_{max} – liczba elementów indeksu przy maksymalnym wypełnieniu

$$W_{min} = 1+2\sum_{i=1}^{h-1}(m+1)^{i-1} = 1+\frac{2((m+1)^{h-1}-1)}{m}$$

$$W_{max} = \sum_{i=1}^{h}(2m+1)^{i-1} = \frac{(2m+1)^h-1}{2m}$$

$$N_{min} = 1+m(W_{min}-1) = 2(m+1)^{h-1}-1$$

$$N_{max} = 2mW_{max} = (2m+1)^h-1$$

B-drzewa – złożoność (c.d.)

B-drzewo klasy t(h,m)

Szacowanie wysokości B-drzewa

- N – liczba elementów w pliku (i w indeksie)
- h_{min} – wysokość przy minimalnym wypełnieniu
- h_{max} – wysokość przy maksymalnym wypełnieniu

$$N_{min} = 2(m+1)^{h-1} - 1$$

$$N_{max} = (2m+1)^{h} - 1$$

$$h_{min} = \log_{m+1}\left(\frac{N-1}{2}\right)$$

$$h_{max} = \log_{2m+1}(N+1)$$

$$h_{max} \leqslant h \leqslant h_{min}$$

$$\log_{2m+1}(N+1) \leqslant h \leqslant \log_{m+1}\left(\frac{N-1}{2}\right)$$

B-drzewa – zadania

1. Oblicz liczbę wierzchołków $\mathbf{W_{min}}$ **i** $\mathbf{W_{max}}$ B-drzewa klasy **T(3, 2)** odpowiednio przy **minimalnym i maksymalnym** wypełnieniu drzewa.

2. Oblicz **maksymalne** (**minimalne**) zużycie pamięci konieczne na zapamiętanie indeksu w postaci B-drzewa, przy następujących danych:

N = 100 000	- liczba rekordów w pliku głównym,
B = 1kB	- wielkość bloku,
P = 4B	- wielkość pola wskaźnikowego,
A = 8B	- wielkość pola adresowego,
X = 20B	- wielkość pola klucza indeksowania.

Jaka jest wówczas **wysokość** B-drzewa?

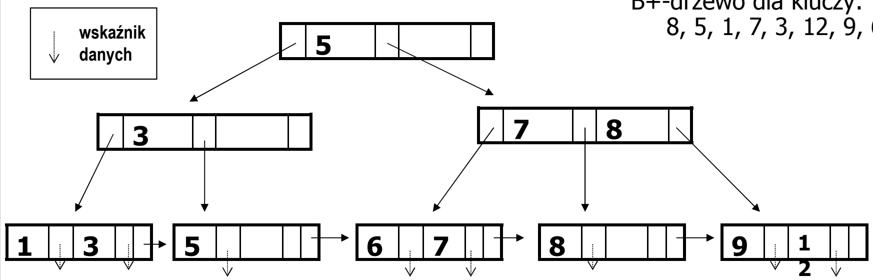
B^+-drzewa

1. W większości komercyjnych systemów baz danych do tworzenia indeksów wykorzystywana jest pewna modyfikacja B-drzew o nazwie **B^+-drzewa.**
2. Struktura B^+-drzewa jest taka sama jak B-drzewa (zrównoważone, wielodrogowe drzewo wyszukiwań).
3. Różnica polega na tym, że wszystkie indeksowane dane i wskaźniki do rekordy danych przechowywane są w **liściach.** Wierzchołki pośrednie służą tylko jako pomoc dotarcia do właściwego liścia.

B^+-drzewa

1. W przypadku B^+-drzewa wskaźniki danych są przechowywane **tylko w liściach**
2. Wierzchołki liści posiadają wpis (indeks ze wskaźnikiem do rekordu danych) dla każdej wartości pola wyszukiwania
3. Niektóre wartości pola wyszukiwania są powtarzane w wierzchołkach wewnętrznych i służą do wspomagania wyszukiwania
4. Koszt wyszukiwania – **wysokość drzewa + 1.**

B+-drzewo dla kluczy:
8, 5, 1, 7, 3, 12, 9, 6



Liście implementowane są zwykle jako lista.

Pytania kontrolne

1. Indeksy: rodzaje indeksów, zastosowania, zalety i wady indeksów, kiedy tworzymy indeksy.
2. B-drzewa: budowa, operacje wstawiania i usuwania elementów, obliczenia (wysokość B-drzewa, ilość wierzchołków, itp.), (przykłady z wykładu).
3. Różnice między B-drzewem a B^+-drzewem.
4. Cele i etapy optymalizacji zapytań, jakie elementy wpływają na koszt wykonania zapytania.

ZADANIA:

1. Jaką wysokość musi mieć B-drzewo klasy **t(h, 5),** aby przy **maksymalnym** wypełnieniu zaindeksować plik liczący **14 500** rekordów?

Pytania kontrolne (c.d.)

1. Dany jest indeks zorganizowany jako B-drzewo klasy **t(3,10)**. Jaka jest **maksymalna** liczba rekordów bazy danych, którą można zaindeksować przy użyciu tego B-drzewa?
2. Strona B-drzewa ma wielkość **1KB**, pole klucza ma długość **16B**, pole wskaźnika na blok ma długość **4B**, pole wskaźnika na rekord ma długość **8B**. Ile indeksów mieści się na **trzecim** poziomie tego drzewa przy **minimalnym** wypełnieniu.

Uwaga: trzeba rozpocząć od obliczenia współczynnika wypełnienia (m) B-drzewa.