Struktury Baz Danych

Miłosz Ilecki

184577

Metoda indeksowania w implementacji

Projekt polegał napisaniu indeksowanej organizacji pliku. W projekcie użyłem organizację indeksowo-sekwencyjną. Metoda ta polega na korzystaniu z dwóch plików. Pierwszy plik jest plikiem indeksowym i zawiera klucz i odpowiadający mu numer strony. Drugi plik zawiera strony z rekordami. Zrealizowałem operacje wstawiania rekordu, odczytu, przeglądanie całej zawartości pliku i indeksu zgodnie z kolejnością wartości klucza oraz operację reorganizacji pliku.

.

Podczas implementacji korzystałem z bibliotek:

- *math* - zaokrąglanie liczb, logarytmy, pierwiastki
- *os*- tworzenie i usuwanie plików,
- *sys*- wyjście z programu,
- *random*- generowanie losowych danych,

Plik z Indeksami

W programie używam pliku z Indeksami. Indeks jest klasą która posiada dwa pola:

- Key - klucz na bazie którego sortujemy (2 bajty)
- PageNo - numer strony na której znajdziemy dany klucz (2 bajty)

Dzięki temu plikowi można szybciej wyszukiwać danych rekordów.

Przykładowy wygląd tego pliku:

Key:	3	PageNumber:	1
Key:	10	PageNumber:	2
Key:	20	PageNumber:	3
Key:	40	PageNumber:	4

Plik z Rekordami

W programie używam pliku z Rekordami. Rekord jest klasą która posiada cztery pola:

- Key - klucz na bazie którego sortujemy (2 bajty)
- Data - zawartość rekordu (4 bajty)
- OverflowPointer - wskaźnik na klucz kolejnego rekordu w OverflowArea (2 bajty)
- Delete - pole czy plik jest usunięty, usunięcie występuję podczas reorganizacji (1 bajty)

Plik z rekordami zawiera strony a na końcu obszar OverflowArea.

Przykładowy wygląd takiego pliku:

Key	Data	Overflow Pointer	Delete
Page number 99			
14	56	15	0
18	123	0	0
20	31	0	0
Overflow Area			
15	3	16	0
16	23	0	0

Specyfikacja pliku

Opisane wcześniej indeksy i rekordy są zapisywane jako bajty.

Indeks zostawał zapisywany na 4 bajtach. Nie stosowano żadnych rozdzielników w celu oszczędności miejsca.

Przykładowo indeks: Key: **5** Pageno : **10** przechowany w formacie bajtów **b'\x00\x05\x00\x0a'**.

Prezentacja wyników

Program ma możliwość wielu sposobów wypisywania.

- print_file() wypisze cały plik ale tylko jako rekordy
- print_record() wypisuje wszystkie strony po kolei i overflow area
- print_indexes() wypisuje plik z indeksami

Eksperyment

W eksperymencie bazowałem na takich stałych:

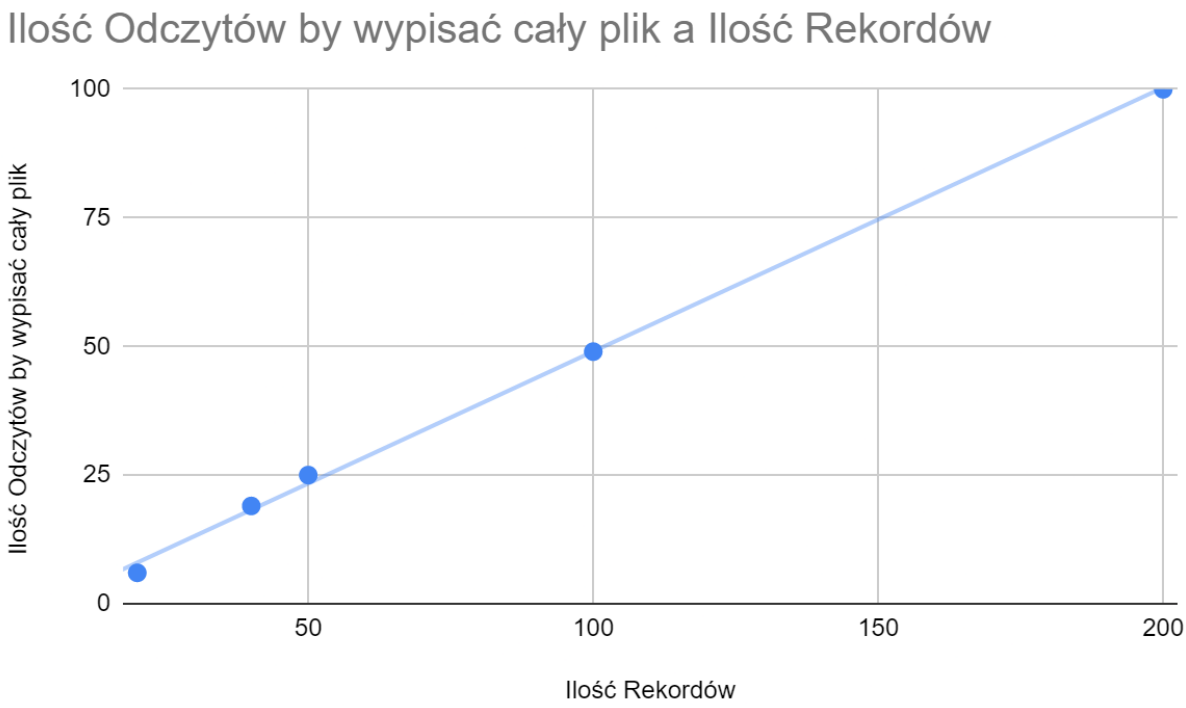
- PAGE_SIZE = 4 maksymalna ilość rekordów na stronę
- RECORD_SIZE = 9 ilość bajtów zajmowana przez rekord
- INDEX_SIZE = 4 ilość bajtów zajmowana przez indeks
- MaxOverflowRecords = PAGE_SIZE maksymalna ilość rekordów w OV
-

Eksperyment 1)

Na różnych ilościach rekordów sprawdzałem ile używamy odczytów i zapisów do plików.

Rekordy były generowane losowo.

Ilość rekordów	Ilość odczytów by wypisać cały plik
20	6
40	19
50	25
100	49
200	100



Jak widać na zaprezentowanym wykresie ilość rekordów liniowo zwiększa ilość odczytów do wypisania pliku.

Eksperyment 2)

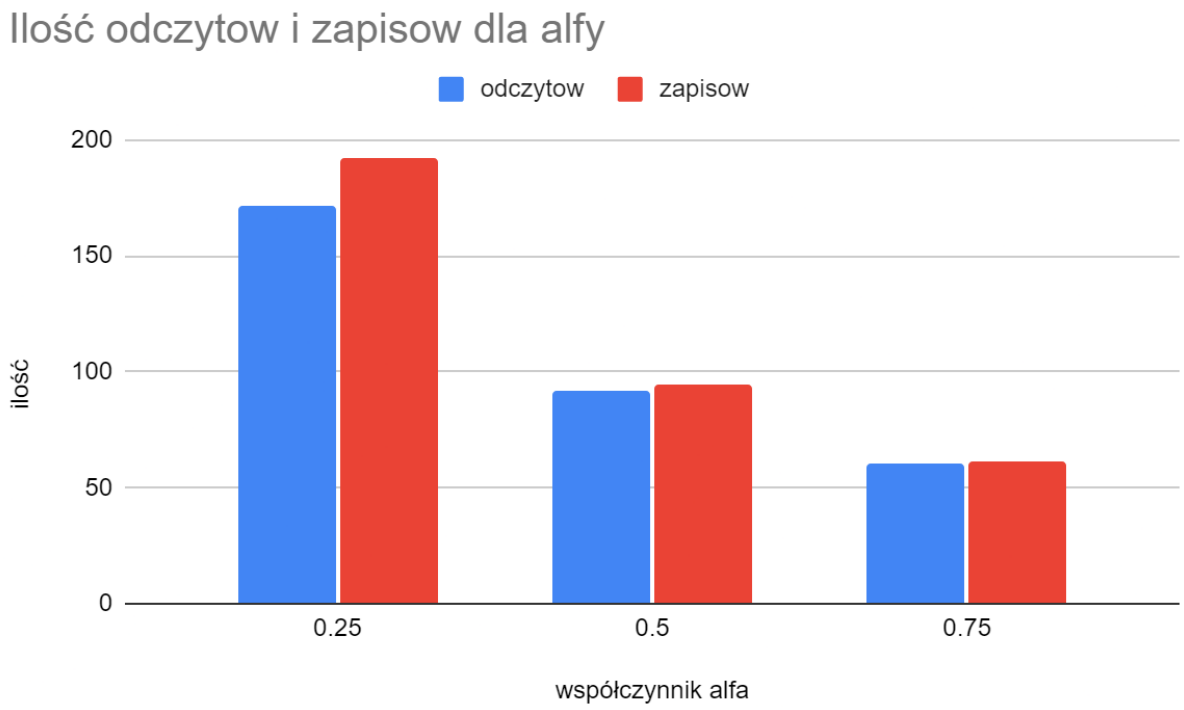
W tym eksperymencie testowałem koszta reorganizacji plików dla różnych współczynników alfa.

ilość odczytów	ilość zapisów
alfa = 0.25	
171	194
176	190
169	186
172	198
średnia	
172	192

alfa = 0.5	
91	92
94	94
90	96
91	96
średnia	
91,5	94,5

alfa = 0.75	
58	58
62	64
61	60
62	64
średnia	
60,75	61,5

Przedstawię teraz średnią ilość odczytów i zapisów dla różnych współczynników alfa.



Z wykresów i danych testowych możemy wywnioskować że współczynnik 0.25 generuje bardzo dużą ilość odczytów i zapisów, w kontrze współczynnik 0.5 redukuje prawie o połowę ich ilość. Współczynnik 0.75 również redukuje ale nie aż w takim dużym przeskoku. Dodatkowo ilość rekordów na stronę wynosiła 4, więc przy współczynniku 0.75 zostaje nam miejsce na jeden rekord, co w krótkim okresie czasu może skutkować zapisaniem czegoś do obszaru nadmiarowego i kolejnej reorganizacji pliku. Reasumując współczynnik 0.5 jest optymalny dla takiego typu danych ze względu na swoich ilość odczytów zapisów, jak i na dłuższy czas zanim nastąpi kolejna reorganizacja.

Eksperyment 3)

W tym eksperymencie testowałem koszt dodania 100 rekordów przy współczynniku alfa = 0.5. Wykonałem 5 serii, średnią z każdej serii pokazuje tabela poniżej. Każda seria to było dodawanie 100 rekordów.

średnia ilość odczytów	średnia ilość zapisów
14.18	5.09
14.34	4.39
12.2	5.35
12.81	5.67
13.48	3.72

Na wyniki odczytów ma wpływ że było odczyt blokowy plików.

Również trzeba uwzględnić że jest to średnia z losowych rekordów. Dodatkowo odczyty i zapisy były naliczane dla rekordu który był dodawany jako pierwszy, i takie co był dodawany jako ostatni czyli wymagał najwięcej operacji.

Interfejs

W programie zaimplementowałem interfejs do przeprowadzania operacji.

Enter a command:

1) add record dodanie nowego rekordu

2) delete record ustawienie rekordu do usunięcia

3) update record aktualizowanie rekordu

4) print whole file wypisywanie całego pliku

5) print records wypisywanie listy rekordów

6) print indexes wypisywanie pliku z indeksami

7) reorganize reorganizacja i realne usuwanie rekordow

8) exit wyjscie z programu

9) input random starting values dodaje 10 losowych wartosci