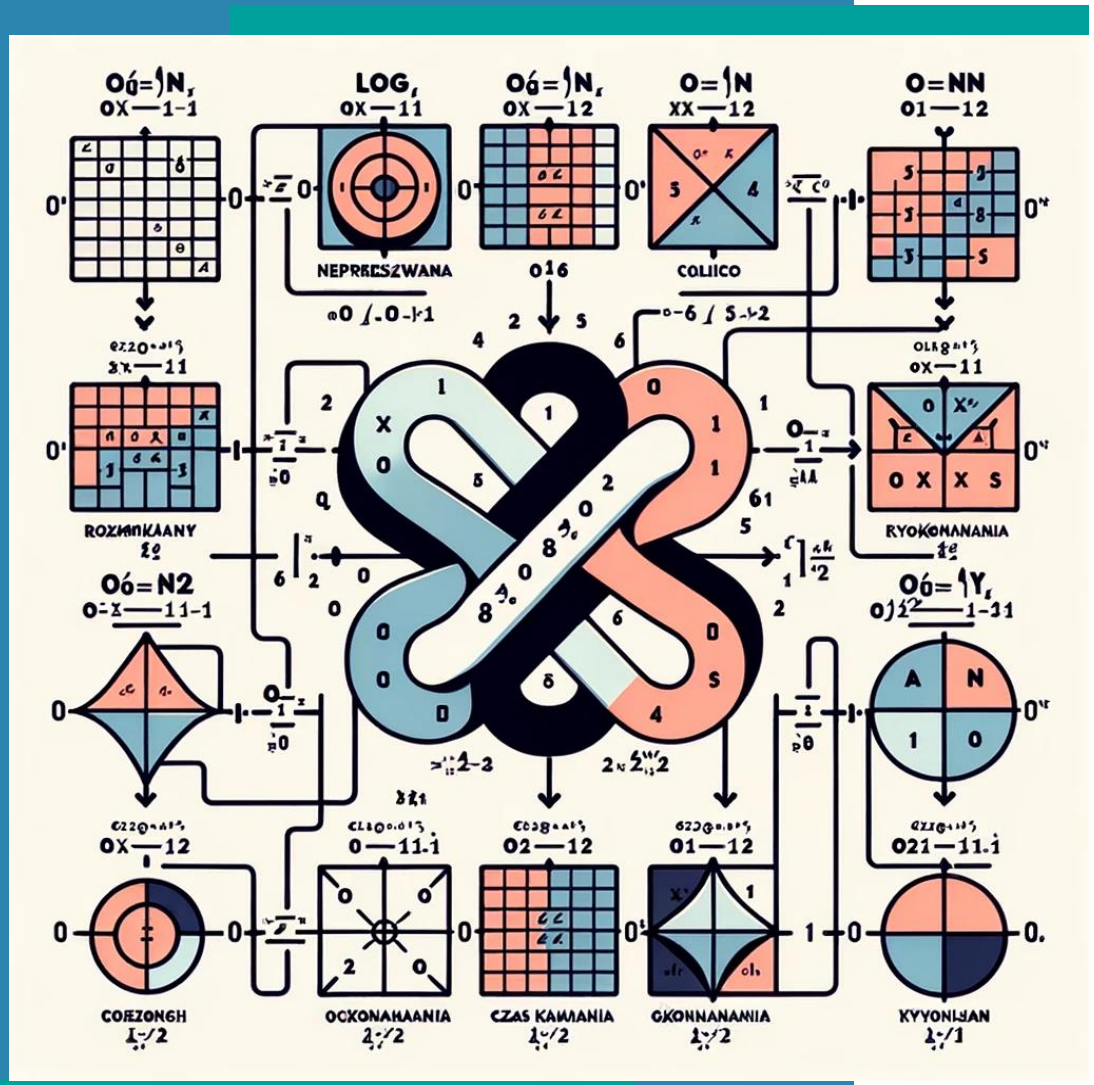
dr inż. Michał Malinowski
michal.malinowski@uth.edu.pl

https://www.linkedin.com/in/michał-malinowski-malkon/

Złożoność obliczeniowa.

Ćwiczenie 02

Algorytmy
i struktury danych



Zagadnienia (4h)

- Złożoność obliczeniowa
  - Złożoność czasowa
  - Złożoność pamięciowa
- Analiza złożoność algorytmów
- Złożoność obliczeniowa kroków algorytmów
  - Zadanie: oszacowanie złożoności czasowej
- Złożoność obliczeniowa instrukcji SQL
  - Zadanie: oszacowanie złożoności czasowej instrukcji SQL
- Zadanie: Złożoność obliczeniowa

Musimy myśleć o programowaniu jak o fundamentalnej umiejętności w XXI w. Porównałbym to do czytania, pisania, liczenia.

Września 2023

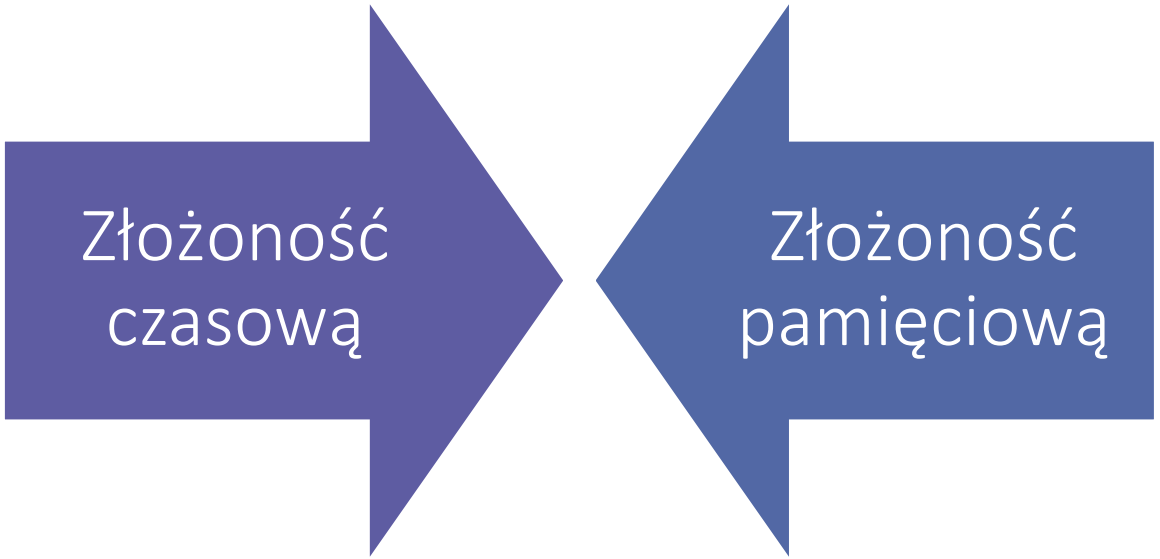
Miron Mironiuk, założyciel Cosmose.AI,
pomysłodawca programu "Programowanie = nasz drugi język"

Złożoność obliczeniowa

Złożoność obliczeniowa

jest to miara określająca zapotrzebowania danego rozwiązania na zasoby systemu w którym ma być rozwiązany

Zasobami branymi pod uwagę są czas i pamięć. Rozróżnia się wiec dwa rodzaje złożoności algorytmów:



Złożoność czasowa

Złożoność czasowa

jest to funkcję opisującą ilość operacji (lub kroków) wykonanych przez algorytm w zależności od wielkości danych wejściowych.

Liczba sekund, godzin, dni .. najczęściej liczba operacji jakie należy wykonać, aby problem został rozwiązany.

Pomaga ocenić, ile czasu zajmuje wykonanie algorytmu w zależności od rozmiaru danych.

Pozwala przewidzieć, jak algorytm zachowuje się w miarę wzrostu ilości danych wejściowych.

Złożoność pamięciowa

Złożoność pamięciowa

odnosi się do ilości pamięci (przestrzeni) wymaganej przez algorytm w zależności od wielkości danych wejściowych.

Liczba bajtów, kilobajtów, megabajtów pamięci RAM i/lub pamięci dyskowej. Również liczba plików, łącz i innych zasobów pamięciowych jakie są niezbędne do realizacji algorytmu.

Pomaga ocenić, ile pamięci zajmuje algorytm.

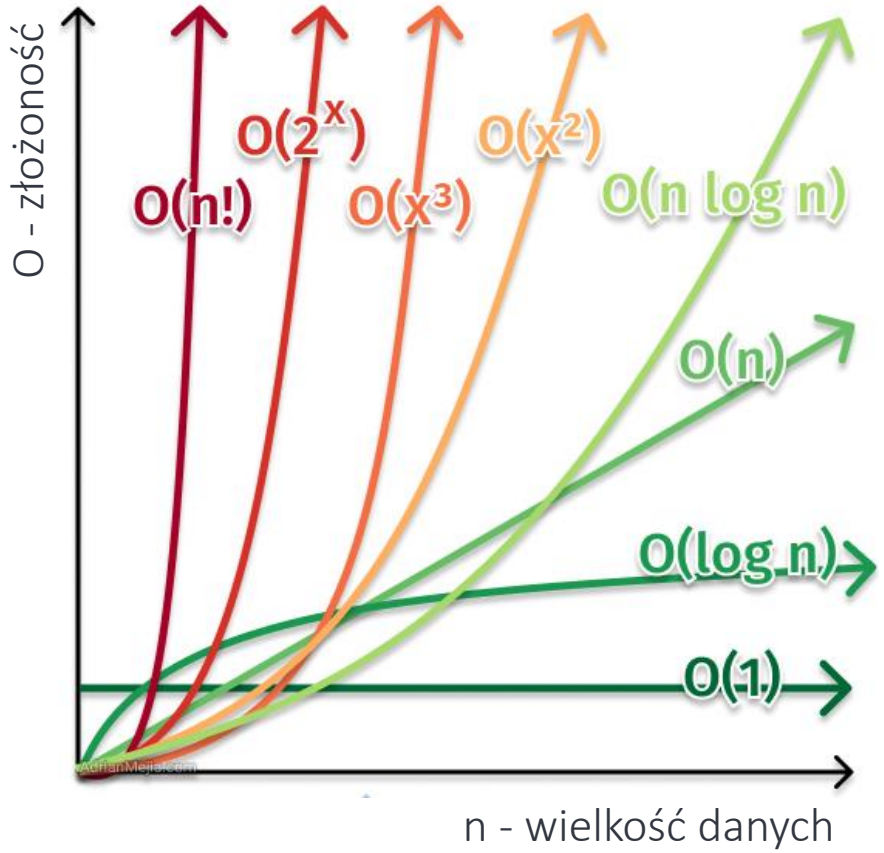
Ważna w sytuacjach, gdzie ograniczenia dotyczą dostępnej przestrzeni pamięci.

Analiza złożoność algorytmów

Notacja O (Wielkie O, Big O)

Notacja O jest matematycznym sposobem opisania górnej granicy złożoności algorytmu w najgorszym przypadku.

Pozwala klasyfikować algorytmy na podstawie ich zachowania względem rozmiaru danych i dzielić na klasy złożoności obliczeniowej.



Analiza złożoność algorytmów

Klasy złożoności obliczeniowe

$O(1)$ - Stała złożoność czasowa:

Oznacza, że czas wykonania algorytmu jest stały i nie zależy od rozmiaru danych wejściowych.
Jest to idealna sytuacja, ale rzadko występuje w praktyce.

$O(n)$ - Liniowa złożoność czasowa:

Oznacza, że czas wykonania algorytmu rośnie liniowo wraz z rozmiarem danych wejściowych.
Przykład: Przeszukiwanie liniowe listy.

$O(\log n)$ - Logarytmiczna złożoność czasowa:

Oznacza, że czas wykonania algorytmu rośnie logarytmicznie wraz z rozmiarem danych wejściowych.
Przykład: Wyszukiwanie binarne w posortowanej liście.

$O(n\ log_n)$ - Złożoność czasowa n log n:

Jest charakterystyczna dla wielu efektywnych algorytmów sortowania.
Przykład: Algorytm QuickSort.

Analiza złożoność algorytmów

Klasy złożoności obliczeniowe

$O(n^2)$ - Kwadratowa złożoność czasowa:

Oznacza, że czas wykonania algorytmu rośnie kwadratowo wraz z rozmiarem danych wejściowych.

Przykład: Sortowanie bąbelkowe.

$O(2^n)$ - Wykładnicza złożoność czasowa:

Oznacza, że czas wykonania algorytmu rośnie wykładniczo wraz z rozmiarem danych wejściowych.

Jest to złożoność, której należy unikać w praktyce ze względu na jej ekstremalną rosnącą złożoność.

Przykład: Obliczenie ciągu Fibonacciego

$O(n!)$- Silnia złożoność czasowa:

Oznacza, że czas wykonania algorytmu rośnie zgodnie z funkcją silnia wraz z rozmiarem danych wejściowych. Jest to najgorszy przypadek i najbardziej nieefektywna złożoność.

Przykład: Problem komiwojażera

Analiza złożoność algorytmów

Porównanie czasów wykonania algorytmów różnych typów przy założeniu, że dla każdy z nich $n = 1$ wykonuje się $10^{-6}s$

n	10	20	30	40	50	60
$O(n)$	0,00001s	0,00002s	0,00003s	0,00004s	0,00005s	0,00006s
$O(n^2)$	0,0001s	0,0004s	0,0009s	0,0016s	0,0025s	0,0036
$O(n^3)$	0,001s	0,008s	0,027s	0,064s	0,125s	0,216s
$O(2^n)$	0,001s	1,048s	17,9min	12,7dni	35,7lat	366w
$O(3^n)$	0,059s	58min	6,5lat	3855w	$227*10^6$w	$1,3*10^{13}$w

Analiza złożoność algorytmów

Reguły upraszczania złożoności obliczeniowej w notacji O

Reguła	Przykład	Uproszczona Forma	Opis
Dominujący Składnik	$O(n^3 + n^2)$	$O(n^3)$	Składnik o najwyższym wpływie na złożoność jest zachowany, reszta jest eliminowana.
Eliminacja Stałych	$O(2n)$	$O(n)$	Stałe mnożniki i dodawane stałe są ignorowane.
Złożoność Logarytmiczna	$O(n^2 + nlog_n)$	$O(n^2)$	Składniki logarytmiczne są mniejsze od składników wielomianowych.
Wielomiany	$O(n^4 + n^3 + n^2)$	$O(n^4)$	Tylko najwyższy stopień wielomianu jest istotny.
Funkcje Złożone i Zagnieżdżone	$O(n \times n)$	$O(n^2)$	Każda zagnieżdżona funkcja jest analizowana i uwzględniana w złożoności.
Funkcje Wykładnicze	$O(2^n + n^2)$	$O(2^n)$	Funkcje wykładnicze są zawsze dominujące wobec innych typów funkcji.
Suma Złożoności	$O(n) + O(n^2)$	$O(n^2)$	Jeżeli algorytm składa się z kilku etapów, ich złożoności są sumowane i następnie upraszczane.
Stała i Liniowa Złożoność	$O(1) + O(n)$	$O(n)$	W przypadku sumy stałej i liniowej złożoności, dominujący składnik to $O(n)$.

Złożoność obliczeniowa kroków algorytmów

Rodzaje kroków algorytmów

Wejście (Input):	O(n)	Wczytywanie n-elementów zazwyczaj ma złożoność liniową w stosunku do liczby elementów.
Wyjście (Output):	O(n)	Wypisywanie n-elementów również ma zwykle złożoność liniową.
Deklaracje (Declarations):	O(1)	Deklarowanie zmiennej lub struktury danych ma zazwyczaj stałą złożoność, niezależnie od wielkości danych.

Złożoność obliczeniowa kroków algorytmów

Rodzaje kroków algorytmów

Przetwarzanie (Processing):	O(1) do O(n^2) lub wyższa	Złożoność przetwarzania zależy od konkretnej operacji. Proste operacje, takie jak dodawanie, mają złożoność O(1), podczas gdy bardziej skomplikowane operacje, takie jak mnożenie macierzy, mogą mieć wyższą złożoność.
Warunki (Conditions):	O(1)	Pojedyncza instrukcja warunkowa (np. if) ma zazwyczaj stałą złożoność, chociaż operacje wewnątrz bloku warunkowego mogą wpłynąć na ogólną złożoność.
Pętle (Loops):	O(n) do O(n^k)	Pętla wykonująca się n razy ma złożoność O(n). Zagnieżdżone pętle mogą mieć wyższą złożoność, na przykład dwie zagnieżdżone pętle mają złożoność O(n^2).
Funkcje (Functions):	Zależy od funkcji	Złożoność funkcji zależy od instrukcji wewnątrz tej funkcji. Funkcje mogą mieć złożoność od O(1) do O(n^k) lub wyższej, w zależności od ich działania.


NIE MOŻESZ PRZESUNĄĆ OBRAZU PALCEM, MUSISZ PODEJŚĆ DO NASTĘPNEGO.
LYNCH

Zadanie: oszacowanie złożoności czasowej

Zadanie

Mamy daną funkcję w języku programowania, która realizuje pewien algorytm.
Zadaniem jest określenie klasy złożoności czasowej tego algorytmu z wykorzystaniem Notacji O.

```
def example_algorithm(arr):
    total = 0
    for i in arr:
        for j in arr:
            total += i * j
    return total
```

Pytania

1. Jaką złożoność czasową ma zmienna komenda total=0?
2. Jaką złożoność czasowa ma każda pętla?
3. Jaka jest ogólna złożoność czasowa

Zadanie: oszacowanie złożoności czasowej

Rozwiązanie

Mamy daną funkcję w języku programowania, która realizuje pewien algorytm.
Zadaniem jest określenie klasy złożoności czasowej tego algorytmu z wykorzystaniem Notacji O.

```
def example_algorithm(arr):
    total = 0                       # O(1)
    for i in arr:                   # O(n)
        for j in arr:               # O(n)
            total += i * j          # O(1)
    return total                    # O(1)
```

Odpowiedzi

1. Jaką złożoność czasową ma zmienna komenda total=0?
   $O(1)$ - stała złożoność czasowa
2. Jaką złożoność czasowa ma każda pętla?
   $O(n)$ - dla każdej z pętli

3. Jaka jest ogólna złożoność czasowa
Złożoność obliczeniowa algorytmu w Notacji O to $O(n^2)$
Po zastosowaniu reguł upraszczania złożoności czasowej otrzymujemy:

$$O(1) + O(n) * O(n) + O(1) + O(1) = O(n^2)$$

Zadanie: oszacowanie złożoności pamięciowej

Zadanie

Zidentyfikowanie i oszacowanie klasy złożoności pamięciowej przedstawionego algorytmu, wyrażonej w Notacji O.

```
Funkcja przykladowy_algorytm(n):
    lista_A = [0, ..., 0]  (rozmiar n)
    lista_B = [0, ..., 0]  (rozmiar n)
    stala x = 0

    Dla i od 0 do n-1:
        Dla j od 0 do n-1:
            x = lista_A[i] + lista_B[j]

    Zwroc x
```

Pytania

1. Jaką złożoność pamięciową ma zmienna x?
2. Jaką złożoność pamięciową mają listy lista_A i lista_B?
3. Jaka jest ogólna złożoność pamięciowa tego algorytmu?

Zadanie: oszacowanie złożoności pamięciowej

Rozwiązanie

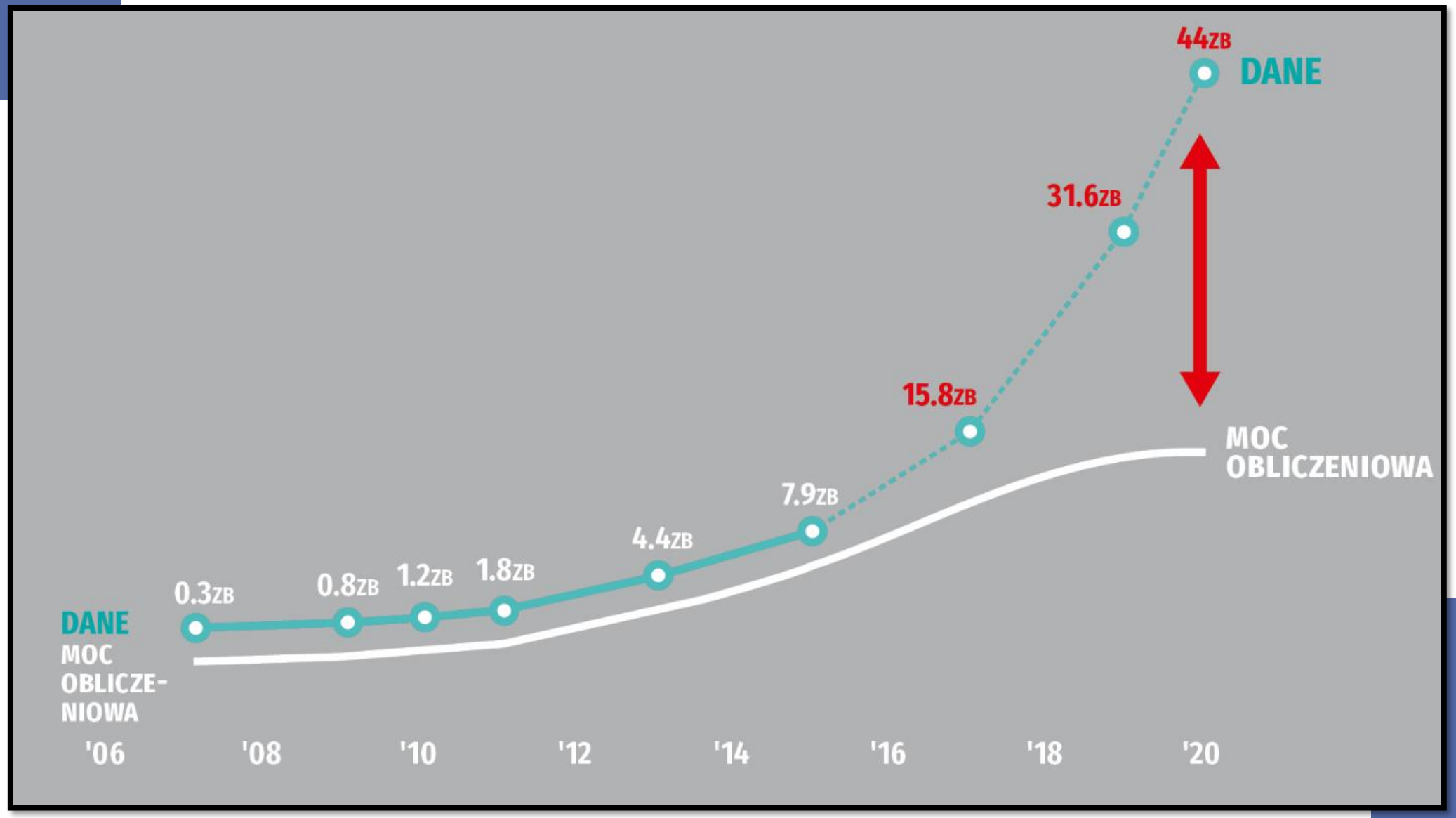
Zidentyfikowanie i oszacowanie klasy złożoności pamięciowej przedstawionego algorytmu, wyrażonej w Notacji O.

```
Funkcja przykladowy_algorytm(n):
    lista_A = [0, ..., 0]  (rozmiar n)
    lista_B = [0, ..., 0]  (rozmiar n)
    stala x = 0

    Dla i od 0 do n-1:
        Dla j od 0 do n-1:
            x = lista_A[i] + lista_B[j]

    Zwroc x
```

Odpowiedzi

1. Jaką złożoność pamięciową ma zmienna x?
   $O(1)$ - stała złożoność pamięciowa dla zmiennej x.
2. Jaką złożoność pamięciową mają listy lista_A i lista_B?
   $O(n)$ - dla każdej z list.
3. Jaka jest ogólna złożoność pamięciowa tego algorytmu?
   $O(1) + 2 \times O(n) = O(n)$ - ogólna złożoność pamięciowa algorytmu.

Porównanie tempa wzrostu ilości danych ze wzrostem mocy obliczeniowej komputerów



Jednostki wielkości danych

Jednostka	Bajty	Wartość dziesiętna w bajtach		Słownie	Przykłady rzeczywistych wielkości
Kilobajt (KB)	1024	1 024	10^3	tysiąc	Mały plik tekstowy lub niewielki obrazek JPEG
Megabajt (MB)	1024^2	1 048 576	10^6	milion	Piosenka w formacie MP3 lub krótki filmik w jakości SD
Gigabajt (GB)	1024^3	1 073 741 824	10^9	miliard	System operacyjny, filmy w jakości HD lub gry komputerowe
Terabajt (TB)	1024^4	1 099 511 627 776	10^12	bilion	Duża baza danych, kolekcja filmów lub zasoby dużego przedsiębiorstwa
Petabajt (PB)	1024^5	1 125 899 906 842 624	10^15	biliard	Zbiory danych w dużych serwerowniach, archiwa bibliotek cyfrowych
Eksabajt (EB)	1024^6	1 152 921 504 606 850 000	10^18	trylion	Całkowita ilość danych generowana przez Internet w ciągu kilku dni
Zettabajt (ZB)	1024^7	1 180 591 620 717 411 303 424	10^21	tryliard	Oszacowana ilość wszystkich danych istniejących w świecie cyfrowym

Złożoność obliczeniowa instrukcji SQL

Rodzaje instrukcji SQL		
SELECT bez JOIN:	Bez indeksu: **O(n)** Z indeksem: **O(log n)**	Wybieranie n rekordów z tabeli ma zazwyczaj złożoność liniową, chociaż indeksy mogą znacząco przyspieszyć operację.
SELECT z JOIN (łączenie dwóch tabel):	Bez indeksu: **O(n * m)** Z indeksem: **O(n log m)** lub **O(m log n)**	Łączenie dwóch tabel o rozmiarach n i m może mieć złożoność kwadratową w najgorszym przypadku. Użycie indeksów i optymalizacje mogą jednak znacznie poprawić wydajność.
INSERT, UPDATE, DELETE:	Bez indeksu: **O(n)** Z indeksem: **O(log n)**	Wstawianie, aktualizowanie lub usuwanie pojedynczego rekordu ma zazwyczaj stałą złożoność, ale jeśli operacje te dotyczą wielu rekordów lub są oparte na pewnych warunkach, złożoność może wzrosnąć do liniowej.

Złożoność obliczeniowa instrukcji SQL

Rodzaje instrukcji SQL

SUM, AVG, COUNT	Bez indeksu: **O(n)** Z indeksem: **O(log n + k)**	Operacje agregujące przeglądają wszystkie rekordy w zbiorze wynikowym. Gdzie k to liczba rekordów spełniających warunek.
MAX, MIN	Bez indeksu: **O(n)** Z indeksem: **O(1)**	
SORT BY (sortowanie):	Bez indeksu: **O(n log n)** Z indeksem: **O(n)**	Sortowanie rekordów zazwyczaj ma złożoność liniowo-logarytmiczną, chociaż optymalizacje mogą przyspieszyć ten proces.
GROUP BY (grupowanie):	Bez indeksu: **O(n log n)** Z indeksem: **O(n)**	Złożoność grupowania zależy od liczby unikalnych kluczy oraz od obecności odpowiednich indeksów.

Zadanie: oszacowanie złożoności czasowej instrukcji SQL

Zadanie

Masz bazę danych składającą się z dwóch głównych tabel:

1. Pracownicy z kolumnami: *ID, Imię, Nazwisko, DataUrodzenia, Stanowisko (indeks)*.
2. Zadania z kolumnami: *ID, Opis, ID_Pracownika, DataPrzydziału, Status (indeks)*.

Pytania

Oszacuj klasę złożoności obliczeniowej następujących zapytań:

1. Wybierz wszystkich pracowników na stanowisku "Programista".
2. Wybierz wszystkich pracowników, którzy mają przypisane zadania o statusie "W trakcie realizacji„
3. Wybierz pracownika z największą liczbą zadań.
4. Wybierz wszystkich pracowników i ich zadania, posortowane według DataPrzydziału.

Zadanie: oszacowanie złożoności czasowej

Rozwiązanie

1. Wybierz wszystkich pracowników na stanowisku "Programista".

   Złożoność: O(log n)

   Ponieważ jest indeks na kolumnie Stanowisko, zapytanie będzie korzystać z indeksu, co przyspieszy wyszukiwanie. Dlatego złożoność jest logarytmiczna.

```
SELECT * FROM Pracownicy
WHERE Stanowisko = 'Programista';
```

Zadanie: oszacowanie złożoności czasowej

Rozwiązanie

2. Wybierz wszystkich pracowników, którzy mają przypisane zadania o statusie "W trakcie realizacji".

   Złożoność: O(n * log m)

   Mimo że jest indeks na kolumnie Status w tabeli Zadania, zapytanie będzie wymagało łączenia tabel, co zwykle ma złożoność liniowo-logarytmiczną.

```sql
SELECT DISTINCT P.ID, P.Imię, P.Nazwisko, P.DataUrodzenia, P.Stanowisko
FROM Pracownicy P
JOIN Zadania Z ON P.ID = Z.ID_Pracownika
WHERE Z.Status = 'W trakcie realizacji';
```

Zadanie: oszacowanie złożoności czasowej

Rozwiązanie

3. Wybierz pracownika z największą liczbą zadań.

   Złożoność: O(n * m)

   Zapytanie będzie wymagało pełnego przeszukiwania obu tabel i agregacji, co wprowadza znaczny koszt. Brak indeksu na kolumnie ID_Pracownika w tabeli Zadania sprawia, że operacja ma złożoność kwadratową.

```sql
SELECT P.ID, P.Imię, P.Nazwisko, COUNT(Z.ID) AS LiczbaZadań
FROM Pracownicy P
JOIN Zadania Z ON P.ID = Z.ID_Pracownika
GROUP BY P.ID, P.Imię, P.Nazwisko
ORDER BY LiczbaZadań DESC
LIMIT 1;
```

Zadanie: oszacowanie złożoności czasowej

Rozwiązanie

4. Wybierz wszystkich pracowników i ich zadania, posortowane według DataPrzydziału.

   Złożoność: O(n * m * log(m))

   Połączenie tabel ma złożoność liniową, ale sortowanie według daty przydziału wprowadza dodatkowy koszt logarytmiczny. Brak indeksu na kolumnie DataPrzydziału zwiększa złożoność sortowania.c

```
SELECT P.ID, P.Imię, P.Nazwisko, Z.ID, Z.Opis, Z.DataPrzydziału, Z.Status
FROM Pracownicy P
LEFT JOIN Zadania Z ON P.ID = Z.ID_Pracownika
ORDER BY Z.DataPrzydziału;
```

Zadanie: Złożoność obliczeniowa

Złożoność obliczeniowa (2 pkt)

Oszacuj złożoność czasową zgodnie z Notacją O dla przedstawionego kodu programu

Uzasadnij słownie swoje oszacowanie.

```
let n = 10;
let m = 20;
let tab_a = new Array(n);
let tab_b = new Array(m);

for (let i = 0; i < n; i++) {
  for (let j = 0; j < n; j++) {
    if (j > i) {
      tab_a[i] = j;
    } else {
      tab_a[i] = i;
    }
  }
}

for (let i = 0; i < m; i++) {
  tab_b[i] = i * i;
}

console.log(tab_a);
console.log(tab_b);
```

Przedstawienie wyniku

- Dokument tekstowy (np.: PDF)
- Umieszczenie wyniku w Moodle

Termin

- Podany przy zadaniu w Moodle (około 2 tygodnie)