# Zagadnienia do zaliczenia ZASD (styczeń, 2024)

## Teoria złożoności algorytmów

1. **Klasa NP i P.**
   - dst: Definicja klasy NP i definicja klasy P.
   - db: Jaka jest różnica pomiędzy problemami z P i NP, czy P=NP?
   - bdb: Problemy której klasy są trudniejsze z P czy z NP? Odpowiedź uzasadnić.

2. **Klasa NP-zupełnych i NP trudnych.**
- dst: Definicja klasy NP-zupełnych i klasy NP-trudnych?
- db: Jaka jest różnica pomiędzy problemami z klas NP-zupełnych i NP-trudnych, czy klasa NP-zupełnych zawiera się w NP-trudnych, a klasa NP-trudnych zawiera się w NP-zupełnych?
- bdb: Problemy której klasy są trudniejsze z NP-zupełnych i czy z NP-trudnych?

3. **Dowodzenie NP-zupełności.**
- dst: Co to jest transformacja wielomianowa problemów, co wystarczy zrobić dla udowodnienia NP-zupełności problemu?
- db: Uzasadnienie, że w ten sposób rzeczywiście można udowodnić NP-zupełność (uzasadnienie ma uwzględnić problem SAT).
- bdb: Znaczenie praktyczne tego, że jakiś problem należy do klasy NP-zupełnych.

4. **Dowodzenie NP-trudności.**
- dst: Co to jest transformacja wielomianowa problemów, co wystarczy zrobić dla udowodnienia NP-trudności problemu?
- db: Wyjaśnić i uzasadnić stwierdzenie: problem NP-trudny to taki problem obliczeniowy, którego rozwiązanie jest co najmniej tak trudne jak rozwiązanie każdego problemu z klasy NP.
- bdb: Znaczenie praktyczne tego, że jakiś problem należy do klasy NP-trudnych.

5. **Twierdzenie Savitcha i jego konsekwencje.**
- dst: Podać treść twierdzenia Savitcha wraz z wyjaśnieniem czego ono dotyczy.
- db: Czego wymaga dowód twierdzenia Savitcha (co trzeba udowodnić)?
- bdb: Jakie jest znaczenie twierdzenia Savitcha?

6. **Twierdzenie Cooka i jego konsekwencje.**
- dst: Podać treść twierdzenia Cooka oraz podać sformułowanie problemu, którego ono dotyczy.
- db: Czego wymaga dowód twierdzenia Cooka (co trzeba udowodnić)?
- bdb: Jakie jest znaczenie twierdzenia Cooka?

7. **Klasa NC (klasa Nicka).**
- dst: Definicja klasy NC.
- db: Czy P=NC?
- bdb: Co daje praktycznie wiedza, że dany problem należy do NC?

## Algorytmy

1. **Równoległe i sekwencyjne sumowanie liczb.**
- dst: Jak realizowane jest sekwencyjne sumowanie liczb i jak jest jego złożoność obliczeniowa czasowa? Jak działa równoległe sumowanie liczb i jak jest jego złożoność obliczeniowa czasowa?
- db: Ile potrzeba procesorów do równoległego sumowania ciągu n liczb? Odpowiedź uzasadnić.
- bdb: Jak liczyć przyspieszenie obliczeń w sumowaniu równoległym w stosunku do sumowania sekwencyjnego, przy czym czas jest mierzony krokami obliczeń, gdzie jeden krok to dodanie 2 liczb

na jednym procesorze? Opowiedzieć o rachunkach jakie trzeba wykonać celem obliczenia przyspieszenia sumowania równoległego 64 liczb na maszynie 32 procesorowej w stosunku do sumowania sekwencyjnego na jednym procesorze. Podać wynik tych rachunków (stosunek liczby kroków w obliczeniach sekwencyjnych do liczby kroków w obliczeniach równoległych).

2. **Sortowanie tablic; porównanie metod: quicksort, mergesort i heapsort**

- dst: Opowiedzieć jak działa sortowanie quicksort, opowiedzieć jak działa sortowanie mergesort, opowiedzieć jak działa sortowanie heapsort
- db: Porównać złożoność obliczeniową czasową i pamięciową wszystkich 3 algorytmów.
- bdb: Przedstawić zalecenia kiedy należy stosować każdą z tych metod w zależności od uwarunkowań danego zastosowania (przesłanki: wiedza o strukturze danych, ryzyko spowolnienia obliczeń, zasoby pamięciowe).

3. **Sortowanie równoległe.**

- dst: Opowiedzieć jak działa sortowanie równoległe metodą naiwną oraz sortowanie równoległe metodą przez scalanie.
- db: Jaka jest złożoność czasowa pamięciowa metody naiwnej i metody przez scalanie? Odpowiedź uzasadnić. Ile procesorów potrzeba w jednej i drugiej metodzie?
- bdb: Porównać obydwie metody w aspekcie praktycznym, tzn. czy i kiedy mogą być stosowane i która jest lepsza w danej sytuacji?

4. **Wyszukiwanie elementów w strukturach danych: liniowe, logarytmiczne, w czasie stałym**.

- dst: Opowiedzieć jak działa wyszukiwanie liniowe elementu w tablicy oraz jak działa wyszukiwanie binarne elementu w tablicy uporządkowanej.
- db: Wskazać jaka struktura konkretna ma wyszukiwanie w czasie stałym i opowiedzieć jak ta struktura jest zbudowana i jak działa (jak się odbywa wstawianie elementów, wyszukiwanie elementów i usuwanie elementów ze struktury).
- bdb: Jaka struktura konkretna danych jest najbardziej efektywna czasowo do wyszukiwania elementów w strukturze gdzie elementy nie są uporządkowane, a jaka gdy elementy są uporządkowane? Odpowiedź uzasadnić przez podanie złożoności wyszukiwania w typowych strukturach konkretnych jak: tablica dynamiczna, lista powiązana, tablica dynamiczna uporządkowana, zrównoważone drzewo BST, drzewo AVL, tablica haszująca.

5. **Sprawdzanie wystąpienia wzorca w tekście; algorytm naiwny N, algorytm Knutha-Morrisa-Pratta, algorytm Karpa-Rabina. Porównanie złożoności i przydatności.**

- dst: Na czym polega problem dopasowania wzorca do tekstu? Opowiedzieć jak działa algorytm naiwny N, algorytm Knutha-Morrisa-Pratta, algorytm Karpa-Rabina. Omówić dwa zastosowania powiązane z problemem dopasowania wzorca do tekstu.
- db: Podać złożoności obliczeniowe czasowe tych algorytmów i wybrać najlepszy algorytm z punktu widzenia złożoności obliczeniowej czasowej.
- bdb: Omówić na czym polega własność algorytmu Karpa-Rabina (albo Knutha-Morrisa-Pratta) sprawdzania wystąpienia wzorca w tekście, która daje prymat (wyższość) temu algorytmowi nad algorytmem naiwnym sprawdzania wystąpienia wzorca w tekście. Na ile własność ta pomaga z punktu widzenia rzędu złożoności obliczeniowej?

6. **Algorytm częściowego dopasowania tekstów algorytmem Needlemana-Wunscha.**

- dst: Na czym polega problem częściowego dopasowania tekstów?
- db: Opowiedzieć jak działa algorytm Needlemana-Wunscha.
- bdb: Omówić trzy zastosowania powiązane z problemem częściowego dopasowania tekstów.

**Struktury danych**

1. **Kopiec zupełny.**

- dst: Co to jest kopiec zupełny? Na czym polega warunek kopca? Jak jest wykonywana operacja wstawiania elementu do kopca i jaką ma złożoność czasową? Jak jest wykonywana operacja usuwania elementu największego z kopca i jaką ma złożoność czasową?
- db: Opowiedzieć jak się odbywa tablicowa implementacja kopca zupełnego.
- bdb: Omówić dwa zastosowania kopca zupełnego (te z wykładu).

2. **Drzewa BST.**

- dst: Co to jest drzewo BST? Na czym polega warunek drzewa BST? Opowiedzieć jak są wykonywane operacje wstawiania elementu do drzewa BST, wyszukiwania elementu w drzewie BST i usuwania elementu z drzewa BST. Jaką złożoność obliczeniową mają te operacje? Odpowiedź uzasadnić.
- db: Opowiedzieć jak się odbywa implementacja drzewa BST.
- bdb: Wyjaśnić na czym polega problem z wyważaniem drzew BST.

3. **Drzewa AVL.**

- dst: Co to jest drzewo AVL? Na czym polega wyższość drzew AVL nad drzewami BST? Opowiedzieć ogólnie jak są wykonywane operacje wstawiania elementu do drzewa AVL, wyszukiwania elementu w drzewie AVL i usuwania elementu z drzewa AVL. Jaką złożoność obliczeniową mają te operacje?
- db: Omówić rotacje pojedynczą i podwójną.
- bdb: Omówić stosowanie rotacji przy wstawianiu i usuwaniu elementu z drzewa AVL.

4. **B-drzewa.**

- dst: Omówić ograniczenia drzew BST i AVL w stosunku do przechowywania danych na dysku. Co to jest B-drzewo? Na czym polega wyższość B-drzew nad drzewami AVL i BST? Jaka jest struktura B-drzewa? Opowiedzieć ogólnie jak są wykonywane operacje wstawiania elementu do B-drzewa, wyszukiwania elementu w B-drzewie i usuwania elementu z B-drzewa. Jaką złożoność obliczeniową mają te operacje?
- db: Omówić podstawowe zastosowanie B-drzew. Różnica pomiędzy B-drzewem a B+-drzewem.
- bdb: Omówić mechanizmy utrzymywania warunku zrównoważenia w B-drzewach.

5. **Trwałe struktury danych.**

- dst: Co to są trwałe struktury danych i jakie mają własności w stosunku do struktur nietrwałych? Jaka jest różnica pomiędzy strukturami ulotnymi, częściowo trwałymi i całkowicie trwałymi? Omówić dwa prymitywne podejścia do tworzenia trwałych struktur danych.
- db: Omówić metodę grubych węzłów, metodę kopiowania ścieżki oraz metoda łącząca metodę grubych węzłów i metodę kopiowania ścieżki.
- bdb: Podać przykład trwałej struktury danych i uzasadnić dlaczego jest ona trwała. Omówić potencjalne zastosowania trwałych struktur danych.