**Algorytmy i Struktury Danych**

**EGZAMIN ONLINE**

**29.06.2021 (wt), godz. 15:00-15:50**

**Nazwisko i imię:** Adrian Stankiewicz

**Nr albumu:** 47576

PUNKTACJA (wypełnia prowadzący)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Zad 1 | Zad 2 | Zad 3 | Zad 4 | Zad 5 | Zad 6 | **RAZEM** | **OCENA** |
|  |  |  |  |  |  |  |  |

1. **Wyjaśnij pojęcia całkowitej i częściowej poprawności algorytmu. Podaj przykład prostego algorytmu, który jest częściowo poprawny, ale nie jest całkowicie poprawny.**

Można powiedzieć, że algorytm jest w pełni poprawny, kiedy jest w pełni zgodny ze swoją specyfikacją, oraz jedynie częściowo poprawny, kiedy jakieś fragmenty specyfikacji nie są spełnione podczas wykonywania algorytmu.

**Przykład:**

Algorytm ma:

A) Przyjąć dwie zmienne typu int

B) Wyliczyć sumę tych 2 liczb

C) Napisać “Brawo!” pod wynikiem.  
D) Zwrócić wynik

Algorytm będzie częściowo poprawny, gdy przyjmie dane, wyliczy ich sumę oraz zwróci wynik, ale nie wypisze “Brawo!”

1. **Czym jest złożoność obliczeniowa algorytmu? Jak ją mierzymy? Podaj złożoność algorytmów wyszukiwania sekwencyjnego oraz binarnego wyszukiwania elementu w posortowanej tablicy.**

Złożoność obliczeniowa algorytmu jest miarą ilości zasobów komputera jaką należy wykorzystać, aby wykonać dany algorytm.  
Wyróżnia się zazwyczaj dwie główne składowe złożoności obliczeniowej:

* Złożoność czasową (zależy od ilości operacji dominujących wykonywanych w trakcie przebiegu algorytmu)
* Złożoność pamięciową (zależy od wielkości danych wejściowych w stosunku do wykorzystanej pamięci)

Obie te złożoności mierzymy przy pomocy funkcji systemowych.

Złożoność przeszukań (przy założeniu, że n to ilość elementów)

* Binarne: O(log n) - operacja dzieli nam listę na fragmenty usuwając wiele niepotrzebnych operacji - LEPSZE
* Sekwencyjne: O(n) - operację wykonujemy tyle razy, ile jest elementów

1. **Wskaż podobieństwa i różnice między podstawowymi strukturami danych: kolejki i stosu. Jak wygląda interfejs obu struktur? W jakie sposób można te struktury implementować?**

Zasada działania:

* Kolejka: FIFO (element, który jest dodany do kolejki jako pierwszy również jako pierwszy ją opuści)
* Stos: LIFO (element, który jako ostatni został położony na stosie jako pierwszy go opuści)

Obie struktury możemy zaimplementować przy pomocy:

* Listy z dowiązaniami
* Tablicy

Interfejs kolejki:

* Konstruktor
* Dodanie elementu na początek
* Pobranie elementu z początku kolejki
* Dostęp do liczby elementów w kolejce
* Sprawdzenie czy kolejka jest pusta

Interfejs

* Konstruktor
* Położenie elementu na górze stosu
* Pobranie elementu z góry stosu
* Pobranie rozmiaru stosu
* Sprawdzenie czy stos jest pusty

1. **Wyjaśnij istotę kolejki priorytetowej uwypuklając różnice w stosunku do podstawowej struktury kolejki? Jaki znasz sposoby jej implementacji? Wskaż zalety i wady każdego z nich.**

Kolejka priorytetowa – Jest to struktura danych pozwalająca na przechowywanie wartości posegregowanych względem jakiegoś priorytetu (np. Wiek osób stojących w kolejce - można ustawić od najmłodszych do najstarszych lub od najstarszych do najmłodszych). W przeciwieństwie do zwykłej kolejki dane które są właśnie do niej dołączane są następnie ustawiane w odpowiedniej kolejności a nie jedynie dostawiane na jej koniec.

Kolejką priorytetową można zaimplementować przy użyciu:

* **Tablicy** *(kolejka zaimplementowana przy użyciu tablicy ma z góry ustawiony rozmiar, który nie może zostać przekroczony, ponieważ tablica jest statycznym typem danych. Jest za to bardzo wydajna zarówno pamięciowo oraz czasowo.)*
* **Listy powiązanej** *(kolejka zaimplementowana przy użyciu listy dowiązań jest w pełni dynamiczna, dzięki czemu można ją dowolnie skracać i rozszerzać. Natomiast jest ona mniej wydajna od tablicy pod względem złożoności czasowej)*
* **Kopca binarnego** *(kolejka zaimplementowana przy użyciu kopca jest z tych trzech opcji najbardziej wydajna. Dzięki wykorzystaniu kopca złożoność czasowa jest najniższa z podanych tutaj struktur przy zachowaniu pełnej możliwości skracania i wydłużania kolejki)*

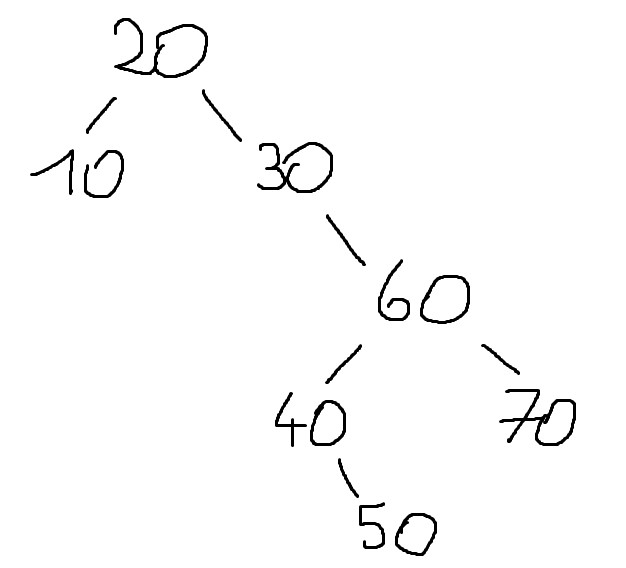
1. **Scharakteryzuj strukturę drzewa wyszukiwań binarnych. Podaj przykład. Jak go można wydajnie reprezentować? Dla podanych kluczy 20, 30, 60, 40, 50, 70, 10 narysuj drzewo, które powstanie po ich wstawieniu w podanej kolejności.**

W drzewie BST zachodzą następujące warunki:

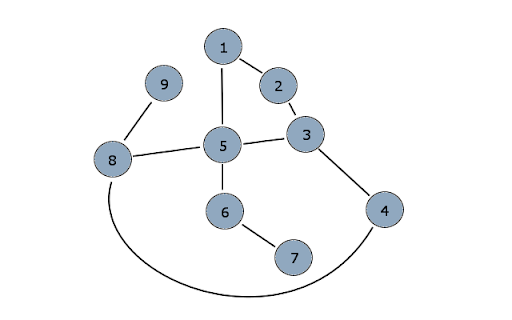
* Wartości na prawo od danego wierzchołka są większe niż on
* Wartości na lewo od danego wierzchołka są mniejsze niż on
* Wierzchołki mają unikalne klucze

Należy też pamiętać, że drzewo BST nie jest samobalansujące więc będzie tworzone odpowiednio z kolejnością dodawania następnych wierzchołków

Drzewo BST można wydajnie zaimplementować oraz reprezentować przy pomocy listy dowiązań, która pozwala nam na dowolne modyfikowanie połączeń między wierzchołkami oraz ilości i wartości samych wierzchołków



1. **Scharakteryzuj strukturę grafu. Wskaż różnice pomiędzy grafem nieskierowanym a skierowanym. Dla poniższego grafu podaj dwa możliwe sposoby jego reprezentowania. Wskaż zalety i wady każdego z nich.**



Graf posiada:

* Wierzchołki
* Krawędzie

Krawędź musi łączyć dwa wierzchołki.

Różnice między grafem nieskierowanym a skierowanym:

* Krawędź w grafie nieskierowanym po prostu łączy dwa wierzchołki i pozwala na poruszanie się po grafie w oba kierunki.
* Krawędź w grafie skierowanym posiada “początek” oraz “koniec” przez co od razu narzuca kierunek, w który trzeba się poruszać po danej krawędzi

Graf można reprezentować za pomocą:

* Macierzy sąsiedztwa: zaletą jest przejrzystość krawędzi i czytelne przedstawienie grafu, jednak wymaga to od nas większych nakładów pamięci
* Tablicy list sąsiedztwa: zaletą jest mniejszy koszt pamięci, jednak, aby czytelnie przedstawić graf wymagane są dodatkowe pętle.