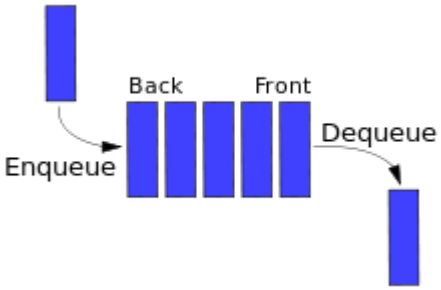
Arkadiusz Kałuża
II – Algorytmy Obsługi Masowej

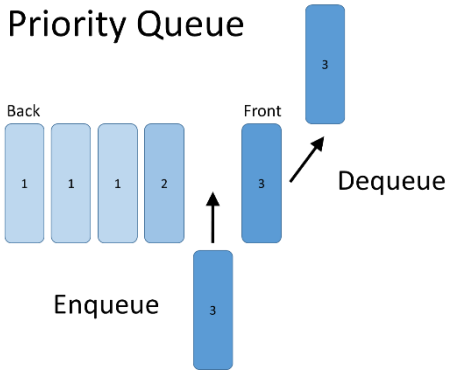
1. **Kolejka** – jest to liniowa struktura danych, w której nowe dane dopisywane są na końcu kolejki, a z początku kolejki pobierane są dane do dalszego przetwarzania. Tak zwane FIFO (first in, first out).
   Dobrym przykładem jest kolejka w sklepie. Im klient później zajmie miejsce w kolejce tym później zostanie obsłużony
   Kolejka zawiera metody takie jak:
   a. Sprawdzanie czy kolejka jest pusta
   b. Sprawdzanie czy kolejka jest pełna
   c. Usunięcie pierwszego elementu z kolejki
   d. Dodanie nowego elementu na koniec kolejki

Idea działania kolejki:



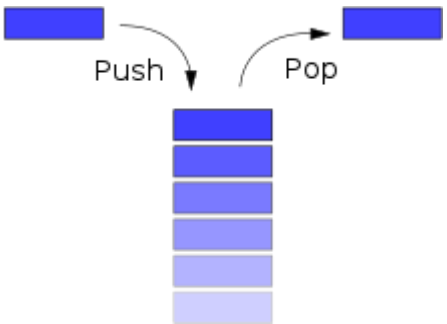
2. **Kolejka z priorytetem** – jest to abstrakcyjny typ danych służący do reprezentowania zbioru elementów, z których każdy ma przyporządkowaną wartość zwaną kluczem, priorytetem i na podstawie tego klucza ustalamy kolejność w kolejce. Najpierw zostaną obsłużone elementy z najwyższym priorytetem, a najpóźniej z najniższym.
   Można to porównać do kolejki u lekarza, czyli pacjenci z najgroźniejszym dla zdrowia/życia urazem zostaną przyjęci jako pierwsi.
   Kolejka piorytetowa zawiera metody takie jak:
   a. Sprawdzanie czy kolejka jest pusta
   b. Sprawdzenie rozmiaru kolejki
   c. Zwrócenie pierwszego elementu z kolejki
   d. Dodanie nowego elementu do kolejki
   e. Usunięcie pierwszego elemetu z kolejki

Idea działania kolejki z priorytetem:



3. **Stos** – jest to liniowa struktura danych, w której dane dokładane są na wierzchu stosu i z wierzchołka stosu są pobierane. Tak zwany LIFO (last in first out). Dobrym przykładem jest tu stos książek, który układamy na biurku, mamy dostęp tylko do książki, która jest na samym wierzchu, a żeby ściągnąć inny musimy ściągnać wszystkie, które są nad nią.
   Stos posiada takie metody jak:
   a. Dodawanie elementu na stos
   b. Pobieranie i usuwanie elementu z wierzchu stosu
   c. Wyświetlanie ostatniego elementu
   d. Wyświetlanie całego stosu
   e. Sprawdzanie czy stos jest pełny
   f. Sprawdzanie czy stos jest pusty

Idea działania stosu:



4. **Lista jednokierunkowa** - strukturą danych, która wykorzystujemy, gdy mamy do czynienia z góry nieznaną ilością danych. Każdy element listy – węzeł (ang. node) zawiera dwa pola. Jedno pole na daną i drugie zawierające wskaźnik na następny element, bez którego po prostu zgubilibyśmy następny element. Pierwszy element listy zwykło nazywać się głową (ang. head), a ostatni ogonem (ang. tail). Wskaźnik ostatniego elementu wskazywać powinien na NULL.
   Implementacje warto podzielić na dwie klasy, a mianowicie klasa opisująca nasz element, węzeł (ang. Node). Który będzie zawierał metody takie jak:
   a. Ustaw nastepny węzeł

b. Zwróć aktualny węzeł

c. Zwróć następny węzeł

Drugą częścię implementacji będzie już klasa opisująca naszą listę jednokierunkową, która będzie zawierać metody takie jak:

a. Dodaj element

b. Zwróć wielkość listy

c. Znajdź element

d. Usuń element

e. Wyświetl elementy listy

Idea działania listy jednokierunkowej:

5. **Lista dwukierunkową** odróżnia od listy jednokierunkowej to, że w każdym elemencie listy jest przechowywane odniesienie zarówno do następnika, jak i poprzednika elementu w liście. Taka reprezentacja umożliwia swobodne przemieszczanie się po liście w obie strony.

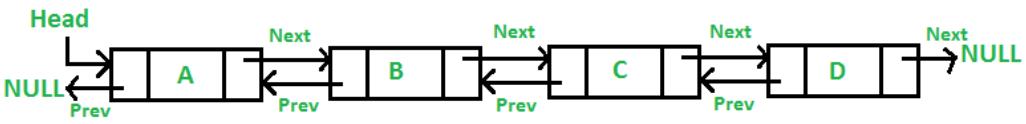
   W implementacji listy dwukierunkowej posłużymy się również klasą Node.
   Lista dwukierunkowa posiada takie metody jak:

   a. Dodaj element z przodu listy

   b. Dodaj element z tyłu listy

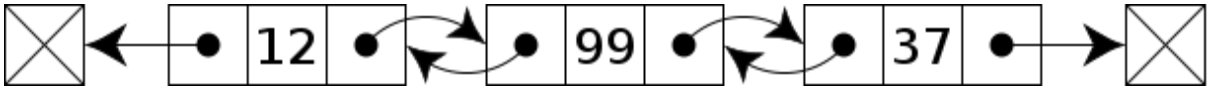
   c. Wyświetl elementy listy

   Idea działania listy dwukierunkowej



6. **Lista cykliczna** różni się tym, że następnikiem ostatniego elementu jest pierwszy element. Po liście można więc przemieszczać się cyklicznie. Nie ma w takiej liście charakterystycznego ogona (ani głowy), często rozpoznawanego po tym, że jego następnik jest pusty (NULL).
   W implementacji listy dwukierunkowej posłużymy się również klasą Node.
   Lista cykliczna posiada takie metody jak:
   a. Zwróć rozmiar listy
   b. Dodaj nowy element do listy
   c. Usuń element z listy
   d. Znajdź element w liście
   e. Wyświetl elementy listy

Idea działania listy cyklicznej:



7. **Lista z wartownikiem(flagą)** – lista z wyróżnionym elementem zwanym wartownikiem. Jest to specjalnie oznaczony element niewidoczny dla programisty wykorzystującego listę. Pusta lista zawiera wtedy tylko wartownika. Zastosowanie wartownika znacznie upraszcza implementację operacji na listach.

   Idea działania listy z flagą: