Lekcja 25 – Kolejka - struktura języka C++

Temat: Omówienie kolejki jako szczególnej struktury danych w językach programowania.

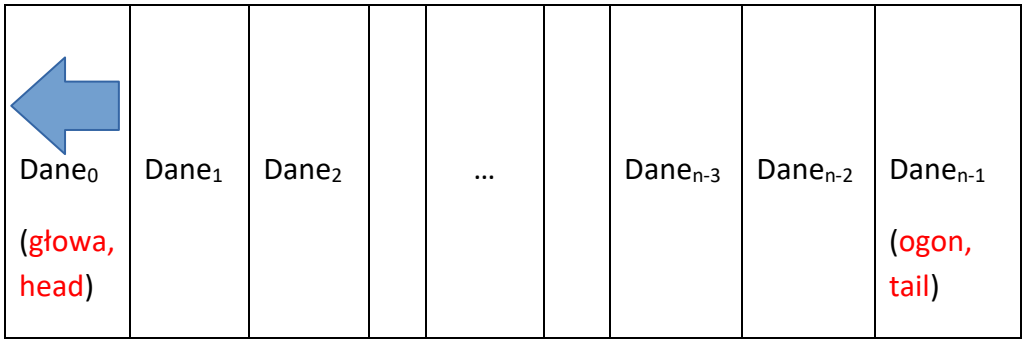
Kod programu źródłowego:

```
/*****************************************************
Kolejka - tablica
*****************************************************/
#include <iostream>
#include <time.h>
using namespace std;

void wyswietl (int *kolejka, int licznik, int rozmiar)
{
 int i;
 for (i=0; i < rozmiar; i++)
 {
 cout << kolejka[i] << " ";
 }
 cout << endl;
}

bool isEmpty (int licznik)
{       //licznik wskazuje na pierwsze puste pole
 if (licznik == 0)
 return true;
 else
 return false;
}

bool isFull (int licznik, int rozmiar)
{       //gdy wskazuje poza kolejka (indeksy kolejkau od 0 do n-1)
 if (licznik == rozmiar)  return true;
 else
 return false;
}
     //Przy wywołaniu funkcji przesyła się wskaźnik do tablicy,
     //chcąc modyfikować zmienną licznik podaje się jej adres w
pamięci
void enqueue (int *kolejka, int a, int &licznik, int rozmiar)
{
    if  (isFull (licznik, rozmiar) == false)
    {
        kolejka[licznik] = a;
        licznik++;
    }
    else
    {
        cout <<"Błąd przepełnienia kolejki \n";
    }
}
   //Przy wywołaniu funkcji przesyła się wskaźnik do tablicy,
```

```
   //chcąc modyfikować zmienne a i licznik podaje się ich adresy w
pamięci
void dequeue (int *kolejka, int &a, int &licznik)
{
    if  (isEmpty (licznik) == false)
    {
        a = kolejka[0];

        for (int i = 0; i < licznik; i++)
        {
            kolejka[i] = kolejka [i+1];
        }
        licznik--;
        kolejka[licznik] = 0;
    }
    else
    {
        cout <<"Błąd! kolejka jest pusta \n";
    }
}

int main()
{
    srand (time (NULL));
    int a, i, temp, rozmiar = 20, licznik = 0;
    int kolejka[rozmiar];

    for (i=0; i < 20; i++)
    {
        kolejka[i]  = 0;
    }
    for (i=0; i < 10; i++)
    {
        a = (rand () % 100) + 1;
        enqueue (kolejka, a, licznik, rozmiar);
    }
    wyswietl (kolejka, licznik, rozmiar);
    for (i=0; i < 5; i++)
    {
        dequeue (kolejka, a, licznik);
    }
    wyswietl (kolejka, licznik, rozmiar);
    return 0;
}
```

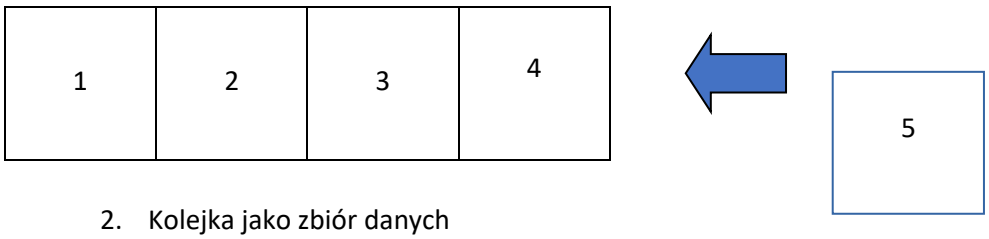
Opis kolejki: Kolejka to struktura danych, w której dane są poukładane liniowo, a dostęp do nich jest możliwy tylko z początku lub końca kolejki.

Kolejka to struktura danych typu FIFO (First-In, First-Out), gdzie pierwsze są pobierane dane, które do kolejki trafiły jako pierwsze.

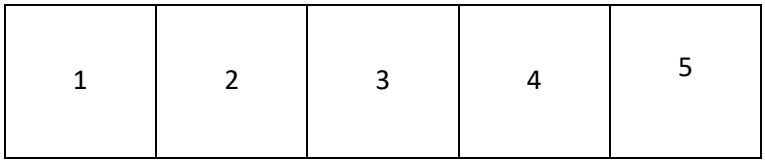


Konsekwencją tej reguły jest to, że elementy są dodawane do kolejki na tylko jednym z jej końców, a usuwane wyłącznie na drugim.

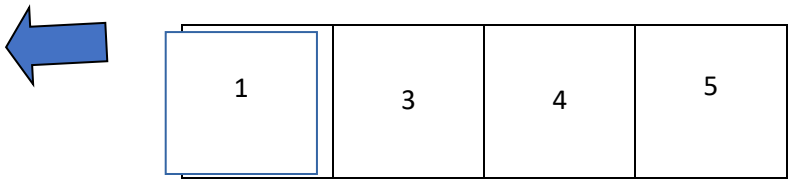
1. Dodanie elementu do kolejki



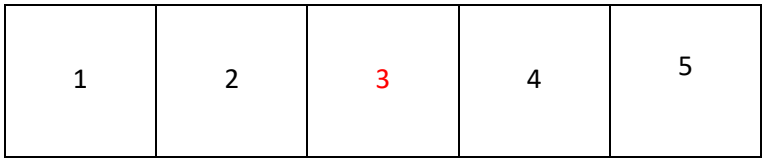
2. Kolejka jako zbiór danych



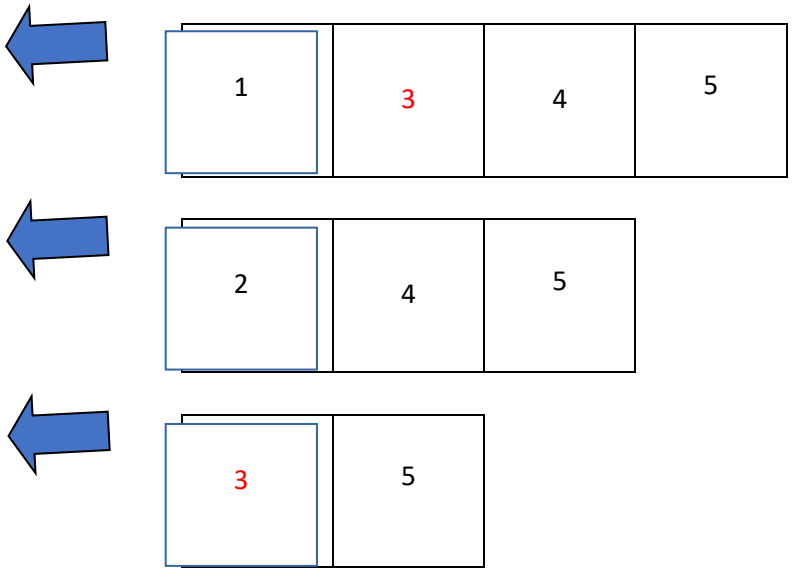
3. Pobranie elementu z kolejki



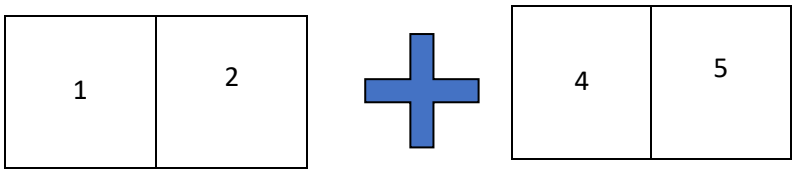
Chcąc dostać się do danych leżących w głębi kolejki, trzeba pobrać wszystkie dane znajdujące się przed nimi.



1. Pobieranie elementów z kolejki



W tym wypadku należy utworzyć nową kolejkę do które dodajemy elementy wcześniejsze. Po odczytaniu szukanych danych, kolejki należy skleić razem.



Zastosowanie:

Kolejkę spotyka się przede wszystkim w sytuacjach związanych z różnego rodzaju obsługą zdarzeń.

W systemach operacyjnych ma zastosowanie kolejka priorytetowa, przydzielająca zasoby sprzętowe uruchomionym procesom.

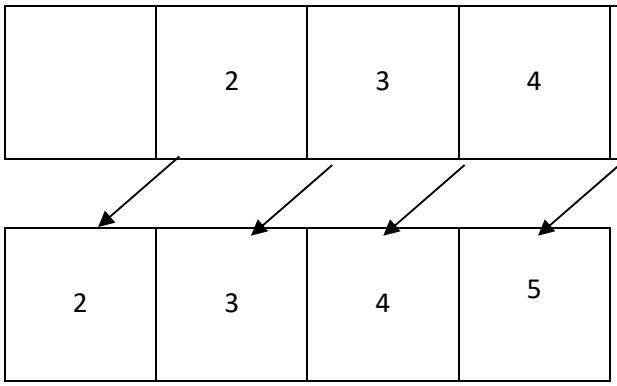
Operacje na kolejce:

- Enqueue (obiekt) (zakolejkuj) - czyli **dodanie obiektu na koniec kolejki**;
- Dequeue () (odkolejkuj) - **pobranie pierwszego obiektu z początku kolejki** i zwrócenie jego wartości;
- isEmpty() – sprawdzenie czy kolejka jest pusta.
- isFull() – sprawdzenie czy kolejka nie jest przepełniona (czy ilość danych nie jest większa niż ilość miejsca na nie).

Implementacja kolejki:

1. Tablicowa.

- Tworzymy tablicę, która będzie kolejką.
- Dostęp do poszczególnych komórek będzie możliwy poprzez zmienną **licznik**, która jest indeksem ostatniego (najwyższego) elementu kolejki. Wskazuje na pierwszy pusty element.
- Pobieramy zawsze element z początku tablicy. Należy potem przesunąć wszystkie elementy od miejsca 1 do końca kolejki o jeden w lewo.



- gdy zmienna **licznik** wskazuje na komórkę 0 (początek kolejki), to próba pobrania nowego elementu wyświetli komunikat o pustej kolejce.
- gdy zmienna **licznik** wskazuje na komórkę poza kolejką, to próba dodania nowego elementu wyświetli komunikat o przepełnieniu kolejki.

1. Wskaźnikowa.

- Tworzymy tablicę, która będzie udostępniać miejsce dla kolejki. Nie musi ona obejmować całego zakresu tablicy!
- Tworzy się następnie wskaźniki, które będą wskazywały początek kolejki (p**oczatek**), jej koniec (**koniec**), aktualnie wskazywaną pozycję (**licznik**).
- Dostęp do poszczególnych komórek będzie możliwy poprzez wskaźnik **licznik**, która jest indeksem ostatniego elementu kolejki. Wskazuje na pierwszy pusty element. Chcąc pobrać element skierować **licznik** ona element wcześniejszy.
- Pobieramy zawsze element z początku tablicy. Należy potem przesunąć wszystkie elementy od miejsca 1 do końca kolejki, o jeden do przodu (zmniejszamy ich indeks o 1).
- gdy wskaźnik **licznik** wskazuje na tę samą komórkę co **poczatek**, to próba pobrania nowego elementu wyświetli komunikat o pustym stosie.
- gdy zmienna **licznik** skazuje na tę samą komórkę co **koniec**, to próba dodania nowego elementu wyświetli komunikat o przepełnieniu stosu.

Ćwiczenie:

1. Napisz program wczytujący łańcuch tekstowy i wprowadzający z niego znaki do kolejki.
2. Napisz funkcję zliczającą liczbę elementów umieszczonych w kolejce.