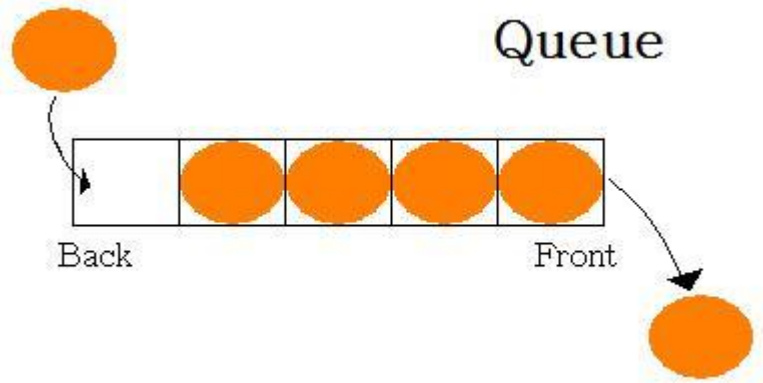
Опашка

доц. д-р Нора Ангелова

Опашка

- Съставна хомогенна линейна структура от данни
- „първи влязъл - пръв излязъл“ (FIFO)



Опашка

Логическо представяне

- Крайна редица от елементи от един и същ тип.

Операции:

- включване - допустима е само за единия край на опашката (край на опашката).
- изключване - допустима е само за другия край на опашката (начало на опашката, глава).

- Пряк достъп - възможен е пряк достъп до елемента, който се намира в началото на опашката.

Опашка

Операции:

- empty() – проверка дали опашката е празна.
- push(x) – включване на елемент в опашката.
- pop() – изключване на елемент от опашката.
- head() – достъп до първия елемент на опашката.

Опашка

Физическо представяне

- последователно – посредством масив
- свързано

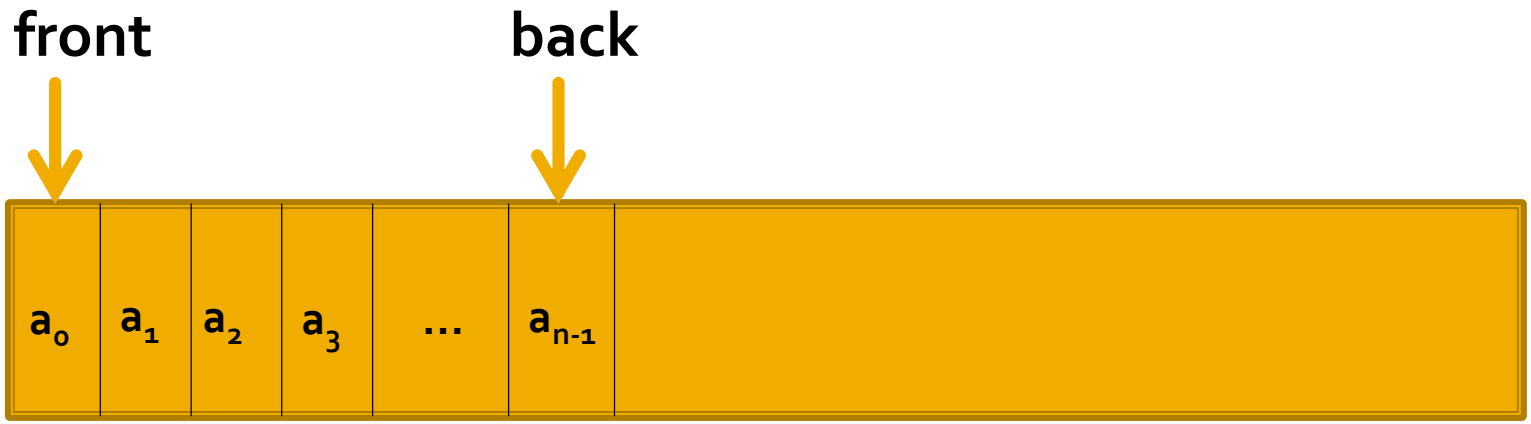
Реализации на опашка

Последователно представяне

- Запазва се блок от памет, в който опашката расте и се съкращава.

Опашка - последователно представяне

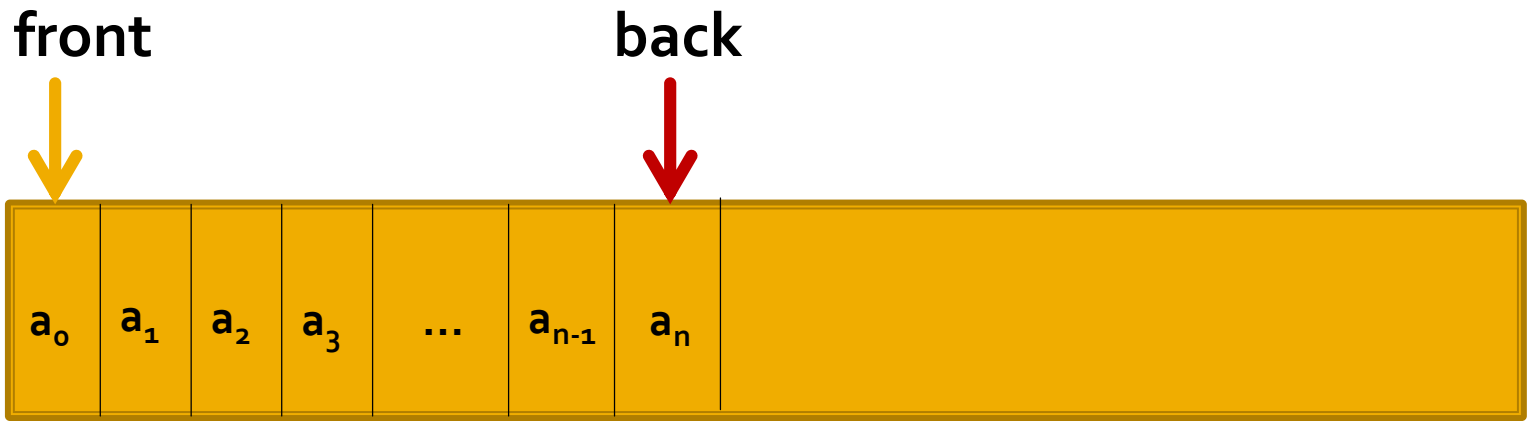
- Статичен масив с ограничен капацитет
  - front – индекс на първия елемент
  - back – индекс на последен елемент на опашката



Опашка - последователно представяне

Реализация с масив

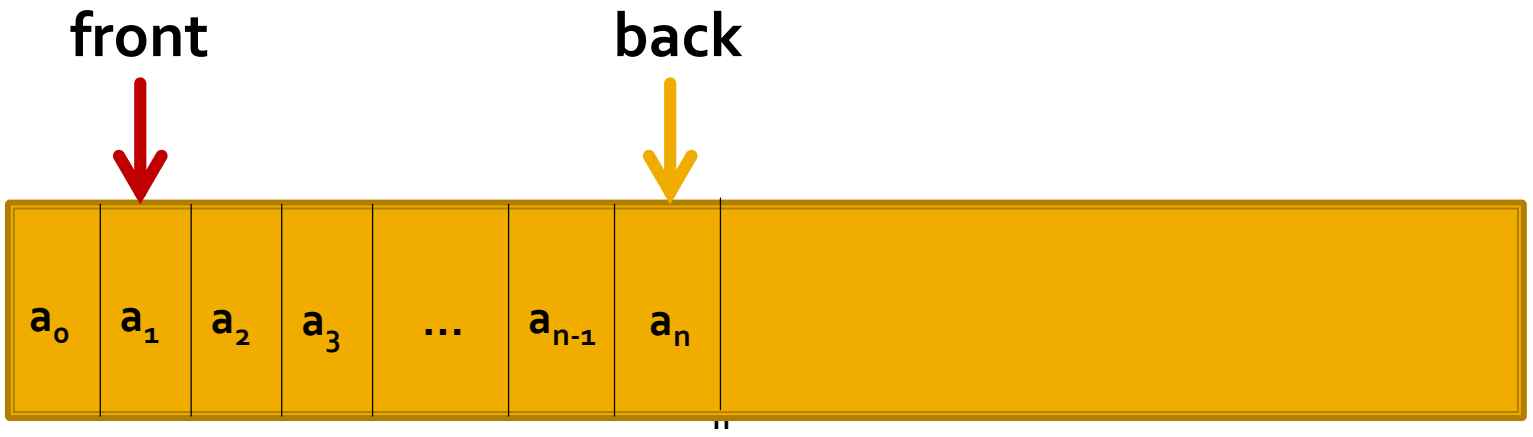
- push ($\mathbf{a_n}$) – включва елемента $\mathbf{a_n}$



Опашка - последователно представяне

Реализация с масив

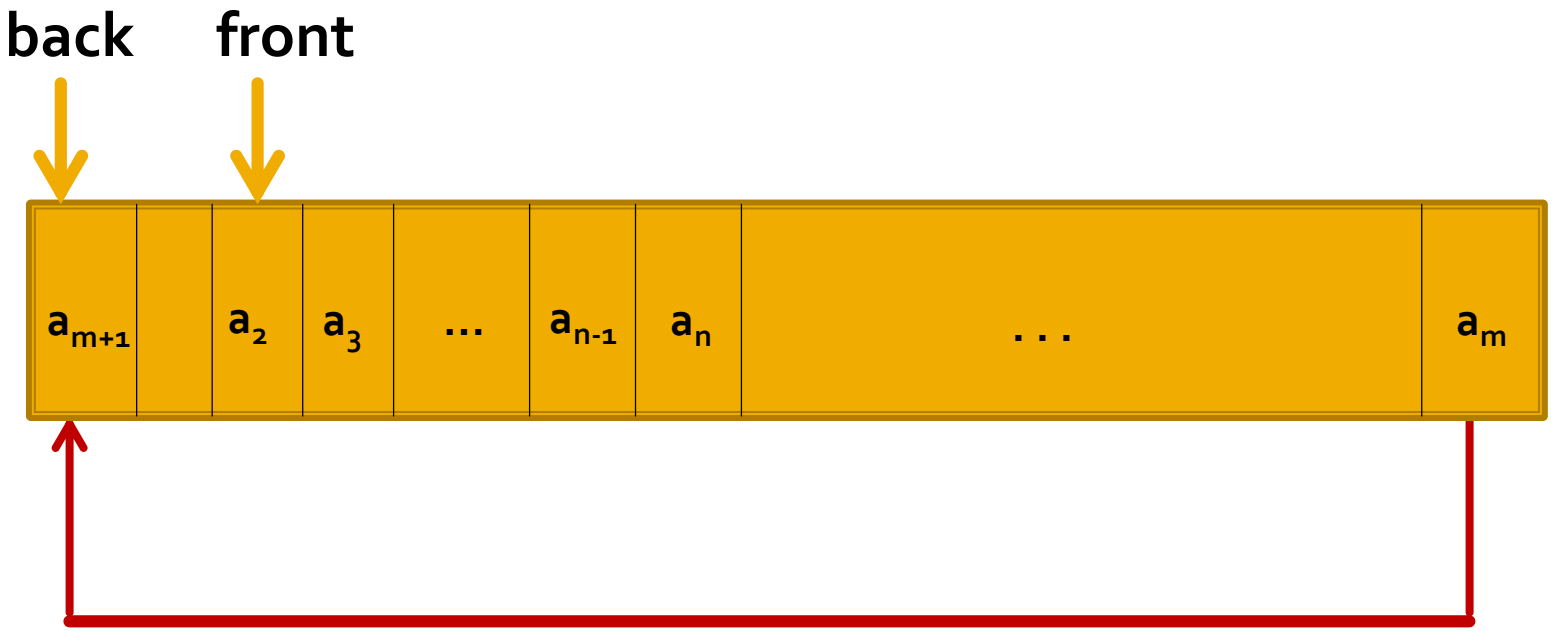
- push ($\mathbf{a_n}$) – включва елемента $\mathbf{a_n}$
- pop () – изключва елемент



Опашка - последователно представяне

Реализация с масив

- push ($\mathbf{a_n}$) – включва елемента $\mathbf{a_n}$
- pop () – изключва елемент
- цикличност

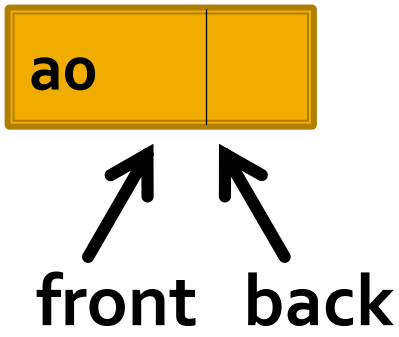


Опашка - последователно представяне

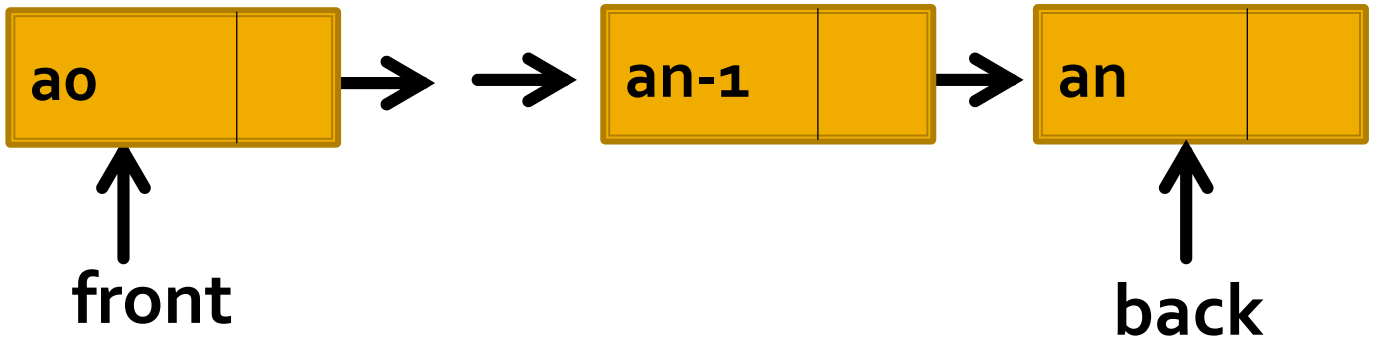
```
const int MAX_SIZE = 1024;

template <typename T>
class StaticQueue {
   T elements[MAX_SIZE];
   // Индекси за начало, край и текущ брой на елементите
   unsigned front, back, size;


   bool full() const;
public:
   StaticQueue();          // Създаване на празна опашка

   bool empty() const;     // Проверка дали опашка е празна
   void push(T const& x); // Включване на елемент
   void pop();             // Изключване на елемент
   T head() const;        // Достъп до първия елемент в опашка
};
```

Опашка – свързано представяне

- Свързано представяне
  - опашка с един елемент



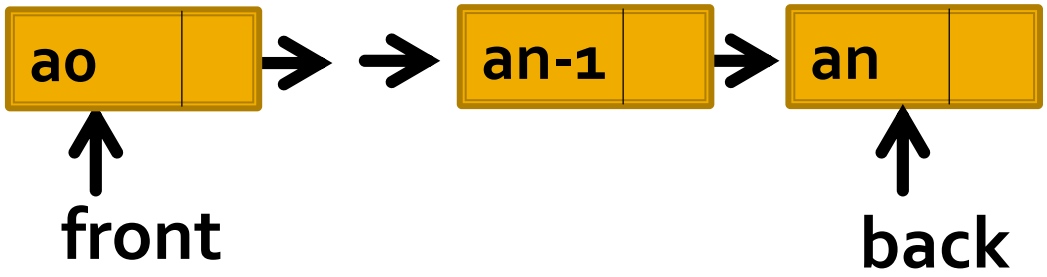
  - опашка с повече от един елемент



Опашка – свързано представяне

Свързано представяне

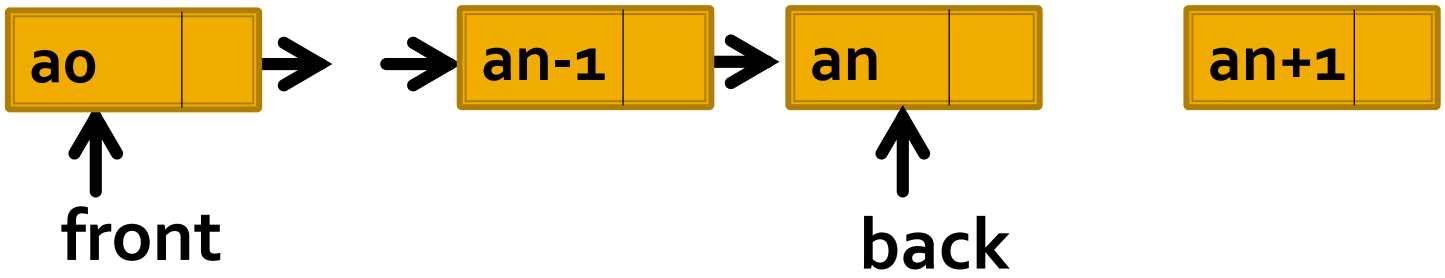
- Добавяне на елемент



Опашка – свързано представяне

Свързано представяне

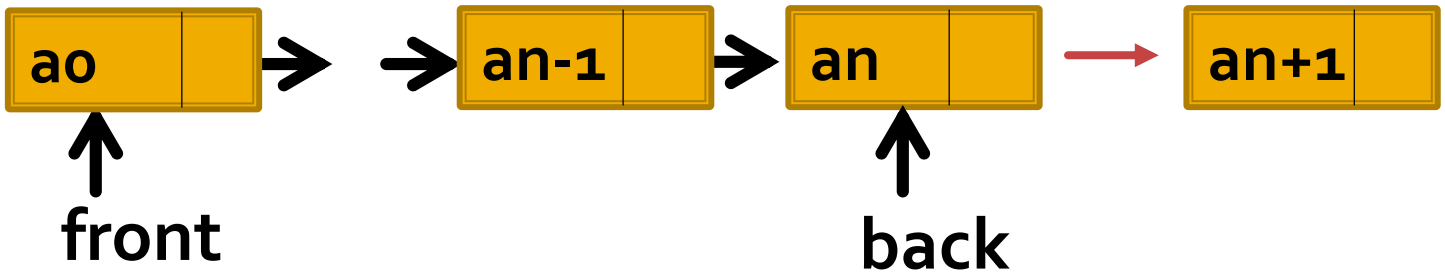
- добавяне на елемент
  - в опашка с повече от един елемент



Опашка – свързано представяне

Свързано представяне

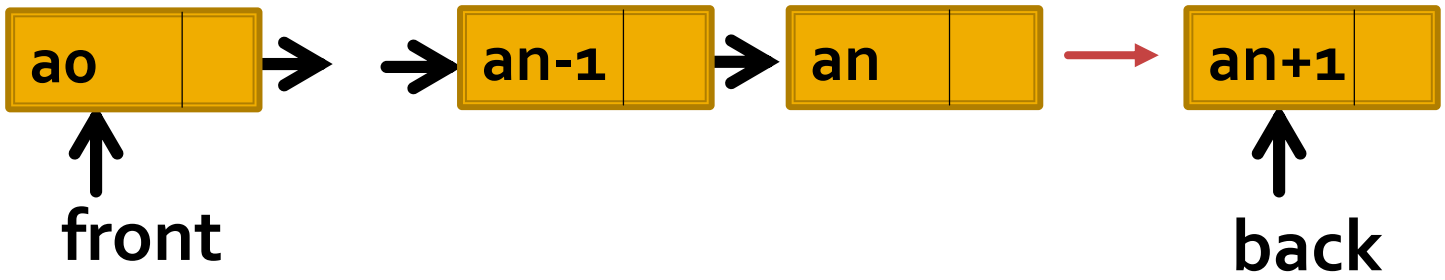
- добавяне на елемент
  - в опашка с повече от един елемент



Опашка – свързано представяне

Свързано представяне

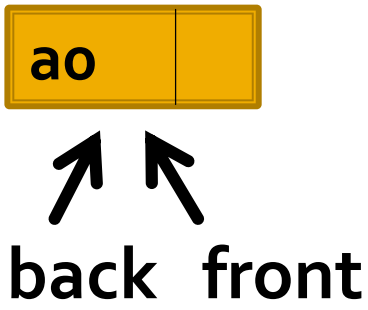
- добавяне на елемент
  - в опашка с повече от един елемент



Опашка – свързано представяне

Свързано представяне

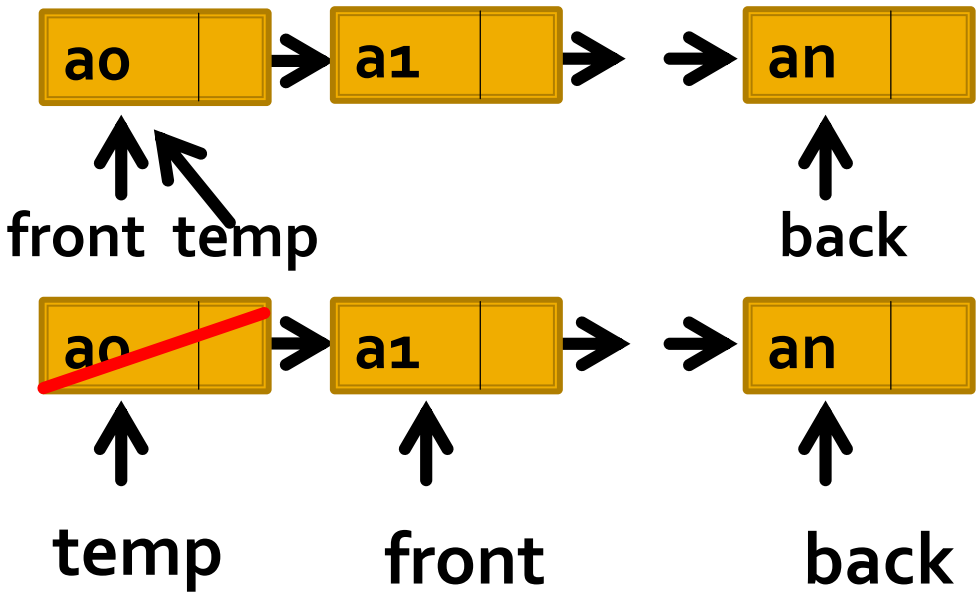
- добавяне на елемент
  - в празна опашка



Опашка – свързано представяне

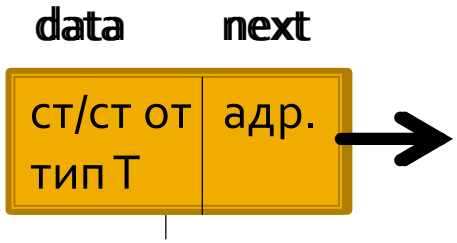
Свързано представяне

- добавяне на елемент
- премахване на елемент



Опашка – свързано представяне

```
template <typename T>
struct QueueElement {
  T data;
  QueueElement<T>* next;
};
```



STL (Опашка)

`std::queue<T>`

#include <queue>

Интерфейс:

- queue() — създаване на празна опашка
- empty() — проверка за празнота на опашка
- push(x) — включване на първи елемент в опашката (void)
- pop() — изключване на последен елемент от опашката (void)
- front() — първи елемент в опашката (reference || const_reference)
- back() — последен елемент в опашката (reference || const_reference)
- size() — дължина на опашката

- ==,!=,,<=,>= — лексикорафско сравнение на две опашки

Следва продължение...