Опашка

Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

Съдържание

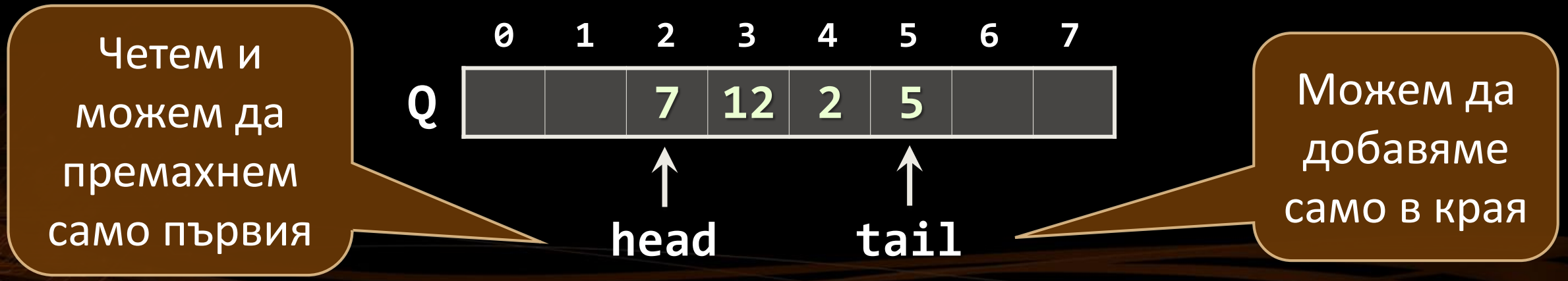
1. Какво е опашка?
2. Статична опашка
3. Динамична опашка
4. Задачи с опашки

Какво е опашка?

- **Опашката** е структура от данни, която има поведение от тип FIFO (First In, First Out) –„първи влиза, първи излиза“.
- **Пример:** опашка в магазин или на светофар; хора на ескалатор; документи, подадени за печат към принтера
- Опашката може да се реализира:
  - **Статично**, чрез масив
  - **Динамично**, чрез възел със стойност и указател към следващ елемент

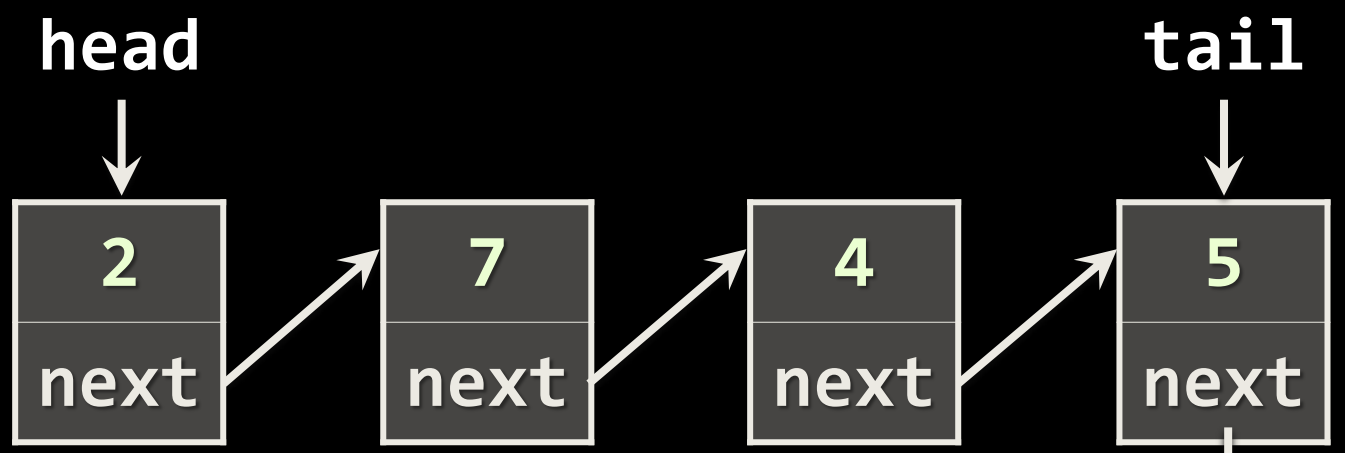
Статична (кръгова) опашка

- Статична (базирана на масив) имплементация
  - Имеплентира се като „кръгов масив“
  - Има ограничен капацитет (като се запълни, се заделя двойно място)
  - Има индекси за начало (**`head`**) и край(**`tail`**), сочещи към началото и края на кръговата опашка



Свързана опашка

- Динамична имплементация
  - Всеки възел (node) има 2 полета: **`value`** и **`next`**
  - Позволява динамично създаване и изтриване



Можем да добавяме само в края

Четем и можем да премахнем само първия

Queue<T> в .NET

- `Queue<T>` имплементира опашка чрез кръгов разтеглив масив
  - Елементите са от един и същ тип `T`
  - `T` може да бъде какъв да е тип, например `int` / `Queue<int>` / `Queue<DateTime>`
  - Размерът се увеличава динамично при нужда

Queue<T>: базова функционалност

- **Enqueue(T)** – добавя елемент в края на опашката

```
queue.Enqueue(5);
```

- **Dequeue()** – премахва и връща елемента от началото

```
int number = queue.Dequeue();
```

- **Peek()** – връща елемента от началото без триене

```
int number = queue.Peek();
```

- **Count** – връща броя елементи в

```
int elementCount = queue.Count;
```

Queue<T>: базова функционалност (2)

- `Clear()` – премахва всички елементи

```
queue.Clear();
```

- `Contains(T)` – проверява дали елемент се среща в опашка

```
bool isFound = queue.Contains(5);
```

- `ToArray()` – преобразува опашка в обикновен масив

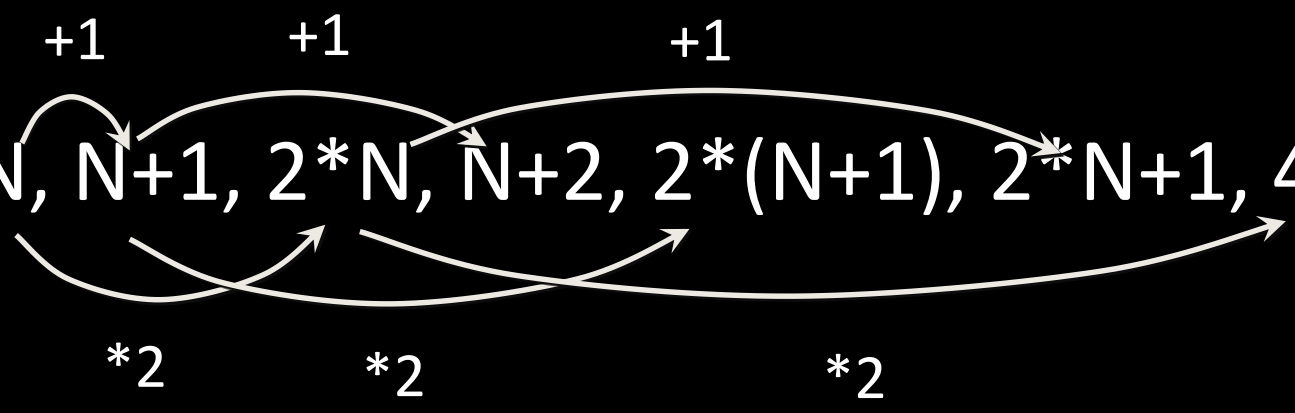
```
int[] arr = queue.ToArray();
```

- `TrimExcess()` – изтрива допълнителното място

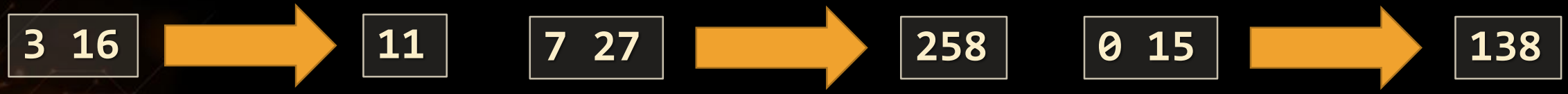
```
queue.TrimExcess();
```

Задача: Редица N, N+1, 2*N ...

- За дадено число N, намерете индекса на елемента от редицата със стойност P. Номерацията започва от 1. Редицата изглежда така:



- S = 3, 4, 6, 5, 8, 7, 12, 6, 10, 9, 16, 8, 14, ...



Решение: Редица N, N+1, 2*N ... (с опашка)

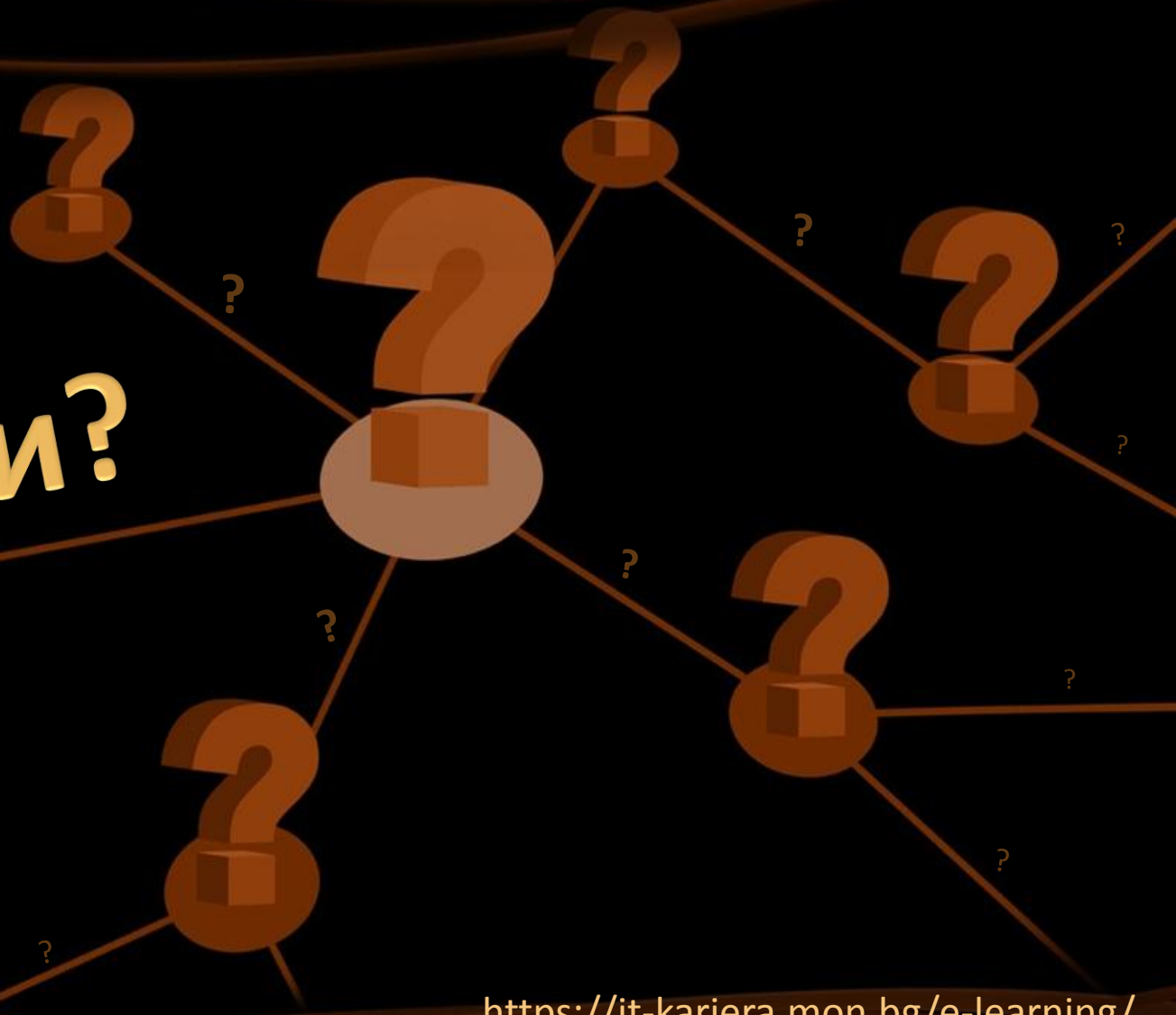
```
int n = 3, p = 16;

Queue<int> queue = new Queue<int>();
queue.Enqueue(n);
int index = 0;
while (queue.Count > 0)
{
    int current = queue.Dequeue();
    index++;
    if (current == p) {
        Console.WriteLine("Index = {0}", index);
        break;
    }
    queue.Enqueue(current + 1);
    queue.Enqueue(2 * current);
}
```

Какво ще се случи, ако за p ни е подадено число, което не присъства в редицата?

Опашка

Въпроси?



https://it-kariera.mon.bg/e-learning/

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**