UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

DISIM
Dipartimento di Ingegneria
e Scienze dell'Informazione
e Matematica

Informatica
@UnivAQ

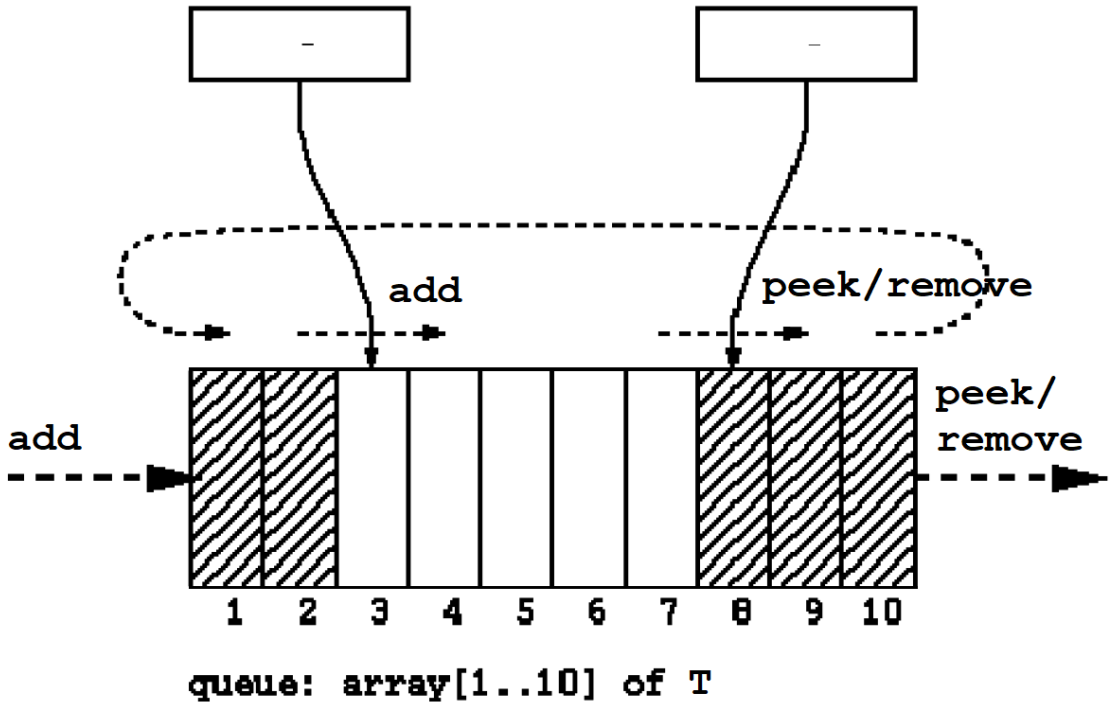
Laboratorio di Algoritmi e Strutture Dati
a.a. 2022/2023

Il tipo di dato Queue

Giovanna Melideo
Università degli Studi dell'Aquila
DISIM

Tipo di dato Coda (Queue)

- Una **coda** è una collezione di elementi dello stesso tipo che supporta le seguenti operazioni tipiche:
  - Enqueue (add), dequeue (remove), peek, isEmpty, [isFull]
- Disciplina di accesso **FIFO - first in first out**: l'accesso agli elementi avviene secondo l'ordine di inserimento



Tipo di dato Queue

tipo `Queue`:

dati: una sequenza S di n elementi.

operazioni:

`isEmpty()` → *result*

restituisce `true` se S è vuota, e `false` altrimenti

`add`(*elem e*)

aggiunge *e* come ultimo elemento di S

`peek()` → *elem // altrimenti riferita come first()*

restituisce il primo elemento di S (senza eliminarlo da S)

`remove()` → *elem*

elimina da S il primo elemento e lo restituisce

Tipo di dato Queue: applicazioni

- Numerose applicazioni delle code in computer science/engineering:
  - **accesso a risorse condivise in mutua esclusione** (coda di accesso alla CPU, spooling di stampa, …)
  - **code di pacchetti** nei dispositivi di rete per **l'instradamento (router)**

L’interfaccia Queue<E> (JCF)

Ogni metodo esiste in due forme:

- Una solleva un’eccezione se l’operazione fallisce
- L’altra restituisce un valore speciale se l’operazione fallisce

	Throws exception	Returns special value
Insert	`add(e)`	`offer(e)`
Remove	`remove()`	`poll()`
Examine	`element()`	`peek()`

L'interfaccia `Queue<E>`

```
public interface Queue<E> extends Collection<E> {
    boolean add(E e)
    E element();
    boolean offer(E e);
    E peek();
    E poll();
    E remove();
}
```

- Le classi `LinkedList<E>` e `PriorityQueue<E>` implementano l'interfaccia `Queue<E>`

L'interfaccia `MyQueue`

- La seguente interfaccia definisce le operazioni di interesse di una coda (**`rif. MyQueue.java`**)

```
public interface MyQueue<T> {
    public boolean isEmpty();
    public boolean offer (T e);
    public T peek();
    public T remove();
    public int size();
}
```

Implementazioni

- Implementazione semplice basata su LinkedList: la coda "delega" banalmente alla lista!

  Rif. LinkedQueue.java

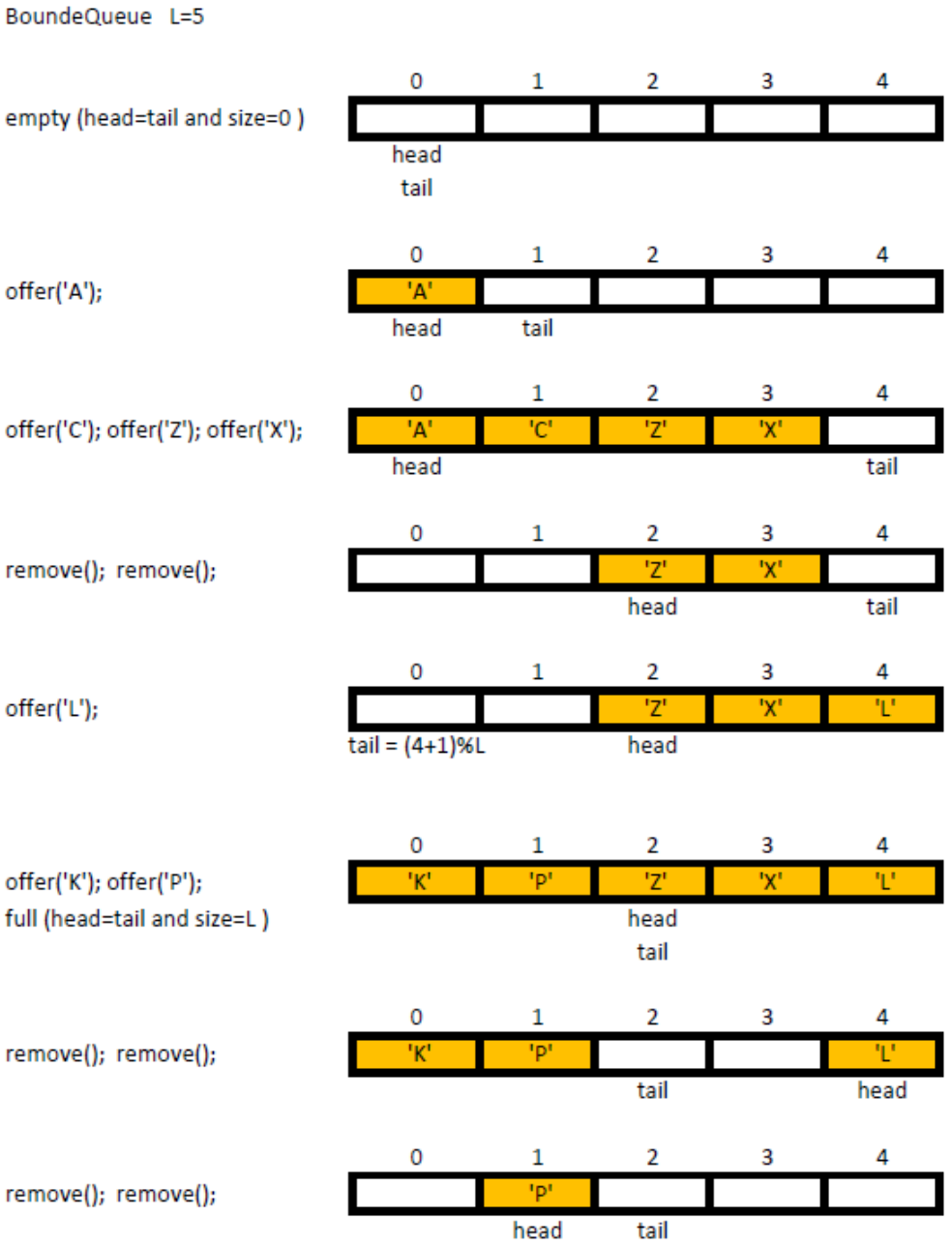
- Implementazione semplice basata su ArrayList: la coda "delega" banalmente alla lista!

  Rif. ArrayListQueue.java

- Implementazione basata su array a dimensione fissa (buffer circolare)

  Rif. BoundedQueue.java

Buffer circolare: esempio



UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

DISIM
Dipartimento di Ingegneria
e Scienze dell'Informazione
e Matematica

Informatica
@UnivAQ

Domande?

Giovanna Melideo
Università degli Studi dell'Aquila
DISIM