

## Laboratorio di Algoritmi e Strutture Dati

Prof. Aniello Murano

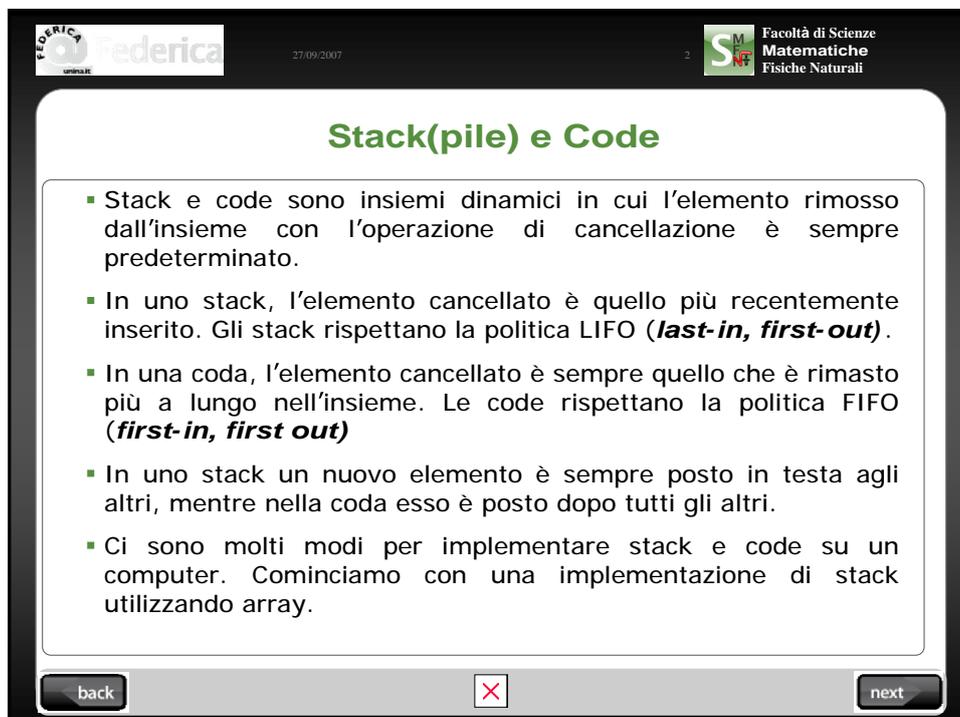
Stack e Code

---

Corso di Laurea Codice insegnamento Email docente Anno accademico	<b>Informatica</b> 13917 murano@na.infn.it 2007/2008	Lezione numero: 5  Parole chiave: <b>LIFO, FIFO</b>
--	---	---

[next](#)




27/09/2007
2 

## Stack(pile) e Code

- Stack e code sono insiemi dinamici in cui l'elemento rimosso dall'insieme con l'operazione di cancellazione è sempre predeterminato.
- In uno stack, l'elemento cancellato è quello più recentemente inserito. Gli stack rispettano la politica LIFO (***last-in, first-out***).
- In una coda, l'elemento cancellato è sempre quello che è rimasto più a lungo nell'insieme. Le code rispettano la politica FIFO (***first-in, first out***).
- In uno stack un nuovo elemento è sempre posto in testa agli altri, mentre nella coda esso è posto dopo tutti gli altri.
- Ci sono molti modi per implementare stack e code su un computer. Cominciamo con una implementazione di stack utilizzando array.

[back](#)
✖
[next](#)

Federica 27/09/2007 3 Facoltà di Scienze Matematiche Fisiche Naturali

## Stack

**Le operazioni fondamentali su Stack sono:**

- **Push(S,x)** che serve a inserire l'elemento **x** al top dello stack
- **Pop(S)** che serve a cancellare il top dello stack **S**

**Un esempio di stack può essere dato da una pila di piatti posta su un tavolo.**

- Si noti che l'ordine in cui i piatti sono tolti dallo stack è inversa all'ordine in cui essi sono stati inseriti nello stack, visto che solo il top dello stack è accessibile.
- Un'altra operazione importante sugli stack è **Empty-Stack**, necessaria per controllare se uno stack è vuoto

back    ✕    next

Federica 27/09/2007 4 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione di Stack con Array

- Per implementare uno stack di **n** elementi, si può utilizzare un array **S** di dimensione **n**.
- L'array **S** utilizzerà l'attributo **TOP** che indicherà l'indice dell'elemento più recente immesso nello stack. In una implementazione si potrebbe pensare di memorizzare questo indice nel primo elemento dell'array che dunque avrà come dimensione non più **n** ma **n+1**.
- Esempio:

	0	1	2	3	4	5
S	4	7	3	2	1	

↑  
Top

- Un'operazione di **Pop** sullo stack dell'esempio precedente, restituirà l'elemento **1** e il **TOP** diventerà **3**.
- Quando **TOP=0**, lo stack è vuoto, cioè non contiene elementi.

back    ✕    next

Federica 27/09/2007 5 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione di Push e Pop

- Utilizzando un array **S[MAX+1]** per l'implementazione di uno stack, nel modo descritto precedentemente, le operazioni di **Push** e **Pop** possono essere realizzate semplicemente nel modo seguente:

**Push**

```
void push(int S[], int valore)
{
    S[0] = S[0]+ 1;
    S[S[0]]= valore;
}
```

**Pop**

```
int pop(int S[])
{
    S[0] = S[0]-1;
    return S[S[0]+1];
}
```

back X next

Federica 27/09/2007 6 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione di Empty\_stack e Full\_stack

- Utilizzando un array **S[MAX+1]** per l'implementazione di uno stack di **MAX** elementi, **empty\_stack** può essere realizzata controllando se **Top=0**. Inoltre, si può utilizzare una procedura **full\_stack** per controllare se lo stack è pieno, cioè se **Top=MAX**.

**empty\_stack**

```
int empty_stack(int S[])
{
    return S[0]==0;
}
```

**full\_stack**

```
int full_stack(int S[])
{
    return S[0]==MAX;
}
```

back X next

Federica 27/09/2007 7 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione efficiente di Push e Pop

- **Facoltativo:** Utilizzando `empty_stack` e `full_stack` è possibile realizzare una versione più efficiente di `Pop` e `Push` con controllo di errore, nel modo seguente:

### Push\_check

```
void push_c(int S[], int val, int *err)
{
    if (full_stack(S))
        *err=1;
    else
    {
        S[0] = S[0] + 1;
        S[S[0]] = valore;
        *err=0;
    }
}
```

### Pop\_check

```
int Pop_c(int S[],int *err)
{
    int val=0;
    if (empty_stack(S))
        *err=1;
    else
    {
        S[0] = S[0]-1;
        val=S[S[0]+1];
        *err=0;
    }
    return val;
}
```

back [X] next

Federica 27/09/2007 8 Facoltà di Scienze Matematiche Fisiche Naturali

## Costruzione di uno Stack

- La seguente procedura permette di costruire uno stack **S** di taglia **num\_elementi**, sapendo che **S** può contenere al più **MAX** valori

```
void new_stack(int S[])
{
    int num_elementi, valore;
    printf("\n Quanti elementi (max %d
elementi): ", MAX);
    scanf("%d",&num_elementi);
    while (num_elementi >MAX) {
        printf("\n max %d elementi: ", MAX);
        scanf("%d",&num_elementi);
    }
    while(num_elementi) {
        printf("\n Inserire un valore: ");
        scanf("%d",&valore);
        push(S, valore);
        --num_elementi;
    }
}
```

- Cosa succede se si inserisce **0** per `num_elementi`?

back [X] next

Federica 27/09/2007 9 Facoltà di Scienze Matematiche Fisiche Naturali

## Stampa di uno Stack

- La stampa di uno stack **S** può essere fatta nel modo seguente:

### Stampa di uno Stack

```
void stampa_stack ( int S[])
{
    int valore;
    while (!empty_stack)
    {
        valore= pop(S);
        printf(" %d ",valore);
        stampa_stack(S);}
    push(S,i);
}
```

back X next

Federica 27/09/2007 10 Facoltà di Scienze Matematiche Fisiche Naturali

## Gestione di uno Stack (1)

- Il seguente programma gestisce uno stack **S** di **MAX** valori. Si noti come i controlli siano indipendenti da MAX. Questo è utile quando MAX e S[MAX+1] sono dati esternamente al programma (come solitamente avviene)

```
#define MAX 20
main()
{
    int S[MAX+1],scelta,valore;
    do
    {
        printf("\n scelta: 0-Crea, 1-Stampa, 2-Pop, 3-Push, 4-uscita : ");
        scanf("%d",&scelta);
        switch (scelta)
            { ..... } /* vedi prossima slide */
    }
    while(scelta==0||scelta==1 ||scelta==2||scelta==3);
} /* fine main() */
```

back X next

Federica 27/09/2007 11 Facoltà di Scienze Matematiche Fisiche Naturali

## Gestione di uno Stack (2): Implement. Switch

```

switch (scelta)
{
    case 0:
        new_stack(S);    break;
    case 1:
        stampa(S);      break;
    case 2:
        if (!empty_stack(S))
            printf("\n Top dello Stack %d", pop(S));
        else
            printf("\n spiacente, stack vuoto");
        break;
    case 3:
        if (!full_stack(S)) {
            printf("\n valore da inserire nello stack: ");
            scanf("%d",&valore);
            push(S,valore);
        }
        else
            printf("\n spiacente, stack pieno");
} /* fine switch */

```

back X next

Federica 27/09/2007 12 Facoltà di Scienze Matematiche Fisiche Naturali

## Esercizio su Stack

- Si consideri uno Stack **S**, implementato con array **S[MAX+1]**.
- Si implementi la funzione **ricorsiva**

**void toglì\_da\_Stack(int S[], int el)**

che elimini dallo stack **S** tutti gli elementi uguali ad **el** lasciando invariato l'ordine degli altri elementi.

- Non utilizzare altre strutture dati di appoggio.
- Non utilizzate accessi diretti all'array, ma servitevi solo delle funzioni implementate per la gestione degli stack.
- Si ricordi che lo stack è una struttura dati che permette l'accesso ai suoi dati solo dal top.

back X next

Federica 27/09/2007 13 Facoltà di Scienze Matematiche Físiche Naturali

## Code

**A differenza dello stack, una coda usa due attributi:**

- Inizio coda (Head)
- Fine coda (Tail)

**In pratica, usando come esempio di coda una fila ad uno sportello. L'inizio della coda è rappresentato dalla prima persona della fila (quella la prossima ad essere servita), mentre la fine della coda è rappresentata dall'ultima persona che si è aggiunta alla coda (cioè, l'ultima persona tra quelle attualmente in fila ad essere servita)**

**Un inserimento nella coda sarà fatto sempre alla sua fine mentre una cancellazione alla sua testa**

back    ✖    next

Federica 27/09/2007 14 Facoltà di Scienze Matematiche Físiche Naturali

## Implementazione di Code

- Come per gli stack, anche per le code possiamo avere diversi modi di rappresentazione su un computer.
- Iniziamo con l'utilizzo di array.
- Per memorizzare una coda con massimo  $n$  elementi, possiamo utilizzare uno stack di dimensione  $n$  più due variabili che memorizzano costantemente l'indice della testa e della coda.
- Come per gli stack, si può anche scegliere di memorizzare l'indice della testa e della coda nel vettore stesso.
- Per esempio:

The diagram shows an array  $S$  with indices 0 through 8. The elements in the array are 3, 2, 1, 7, 5, followed by two empty slots. The **Head** pointer is at index 0, pointing to the value 3. The **Tail** pointer is at index 8, pointing to the value 7. This illustrates how the queue is implemented using an array with pointers to the start and end of the current elements.

back    ✖    next

Federica 27/09/2007 15 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione di Empty\_queue e Full\_queue

- Utilizzando un array  $Q[\text{MAX}+2]$  per l'implementazione di una coda di  $\text{MAX}$  elementi, **empty\_queue** può essere realizzata controllando se  $\text{Head}=0$  (in questo caso  $\text{Teal}$  sarà uguale a 1).
- Inoltre, si può utilizzare una procedura **full\_queue** per controllare se la coda è piena, cioè se  $\text{Head}=\text{Tail}$ .

**empty\_queue**

```
int empty_stack(int Q[])
{
    return Q[0]==0;
}
```

**full\_queue**

```
int full_queue(int Q[])
{
    return Q[0]==Q[MAX+1];
}
```

- Dunque, per creare una coda vuota basterà avere  $\text{Head}=0$  e  $\text{Teal}=1$

back X next

Federica 27/09/2007 16 Facoltà di Scienze Matematiche Fisiche Naturali

## Implementazione di Dequeue e Enqueue

Usando l'array  $Q$  precedente per l'implementazione di una coda, le operazioni di Cancellazione (Dequeue) e Inserimento (Enqueue) possono essere realizzate come segue:

**Enqueue**

```
void enqueue(int Q[], int valore)
{
    Q[Q[MAX+1]] = valore;
    if (Q[0] == 0)
        Q[0]=1;
    Q[MAX+1] = (Q[MAX+1] % MAX) + 1;
}
```

**Dequeue**

```
int dequeue(int Q[])
{
    int valore=Q[Q[0]];
    Q[0] = (Q[0] % MAX) + 1;
    if (Q[0] == Q[MAX+1]) {
        Q[0]=0;
        Q[MAX+1]=1;
    }
    return valore;
}
```

back X next

Federica 27/09/2007 17 Facoltà di Scienze Matematiche Fisiche Naturali

## Stampa di una Coda

- Per la stampa di una coda, non possiamo usare esattamente la stessa procedura vista per gli stack. Infatti questa produrrebbe una inversione della coda. Per risolvere questo problema, provvediamo a invertire ulteriormente la coda.

### stampa\_queue

```
void stampa_queue(int Q[])
{
    int val;
    while (!empty_queue(Q))
    {
        val=dequeue(Q);
        printf(" %d \\", val);
        stampa_queue(Q);
        enqueue(Q, val);
    }
    reverse(Q)
}
```

### reverse

```
void reverse (int Q[])
{
    int val;
    while (!empty_queue(Q))
    {
        val=dequeue(Q);
        reverse(Q);
        enqueue(Q, val);
    }
}
```

back X next

Federica 27/09/2007 18 Facoltà di Scienze Matematiche Fisiche Naturali

## Costruzione di una Coda

- La seguente procedura permette di costruire una coda **Q** di taglia **num\_elementi**, sapendo che **Q** può contenere al più **MAX** valori

```
void new_queue(int Q[])
{
    int num_elementi, valore;
    printf("\n Quanti elementi (max %d elementi): ", MAX);
    scanf("%d", &num_elementi);
    while (num_elementi > MAX) {
        printf("\n max %d elementi: ", MAX);
        scanf("%d", &num_elementi);
    }
    while(num_elementi) {
        printf("\n Inserire un valore: ");
        scanf("%d", &valore);
        enqueue(Q, valore);
        num_elementi--;
    }
}
```

back X next

Federica 27/09/2007 19 Facoltà di Scienze Matematiche Fisiche Naturali

## Gestione di una Coda (1)

- Il seguente programma gestisce una coda **Q** di **MAX** valori. Si noti come i controlli siano indipendenti da MAX. Questo è utile per le stesse motivazioni date per gli stack

```
#define MAX 20
main()
{
    int Q[MAX+1],scelta,valore;
    do
    {
        printf("\n scelta: 0-Crea, 1-Stampa, 2-Deq, 3-Enq,, 4-uscita : ");
        scanf("%d",&scelta);
        switch (scelta)
            { ..... } /* vedi prossima diapositiva */
    }
    while(scelta==0||scelta==1 ||scelta==2||scelta==3);
} /* fine main() */
```

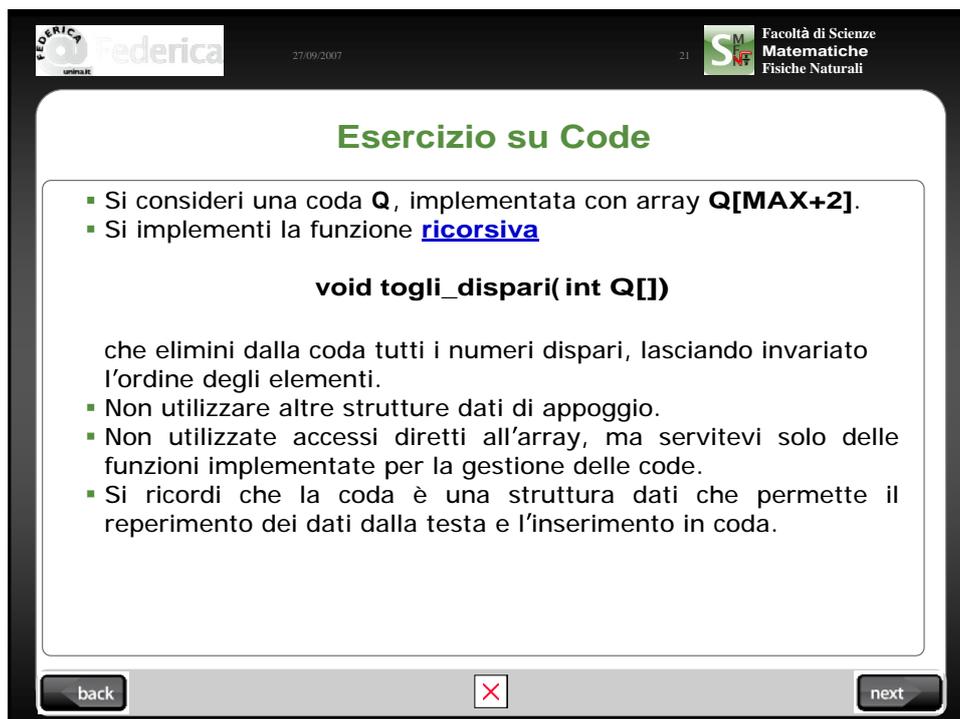
back ✖ next

Federica 27/09/2007 20 Facoltà di Scienze Matematiche Fisiche Naturali

## Gestione di una Coda(2): Implement. switch

```
switch (scelta)
{
    case 0:
        new_queue(Q);          break;
    case 1:
        stampa_queue(Q);      break;
    case 2:
        if (!empty_queue(Q))
            printf("\n Head della coda %d", dequeue(S));
        else
            printf("\n spiacente, coda vuoto");
        break;
    case 3:
        if (!full_queue(Q))    {
            printf("\n Valore da inserire nello stack: ");
            scanf("%d",&valore);
            enqueue(Q,valore);
        }
        else
            printf("\n spiacente, coda piena");
} /* fine switch */
```

back ✖ next



The image shows a presentation slide with a black border. At the top left, there is a logo for 'Federica' with the URL 'www.federica.it' and the date '27/09/2007'. At the top right, there is a logo for the 'Facoltà di Scienze Matematiche Fisiche Naturali' with the number '21'. The main content of the slide is centered and contains the following text:

### Esercizio su Code

- Si consideri una coda **Q**, implementata con array **Q[MAX+2]**.
- Si implementi la funzione **ricorsiva**

**void toglì\_dispari(int Q[])**

che elimini dalla coda tutti i numeri dispari, lasciando invariato l'ordine degli elementi.

- Non utilizzare altre strutture dati di appoggio.
- Non utilizzate accessi diretti all'array, ma servitevi solo delle funzioni implementate per la gestione delle code.
- Si ricordi che la coda è una struttura dati che permette il reperimento dei dati dalla testa e l'inserimento in coda.

At the bottom of the slide, there are three buttons: 'back' on the left, a red 'X' icon in the center, and 'next' on the right.

This document was created with Win2PDF available at <http://www.win2pdf.com>.  
The unregistered version of Win2PDF is for evaluation or non-commercial use only.  
This page will not be added after purchasing Win2PDF.