# Grafi

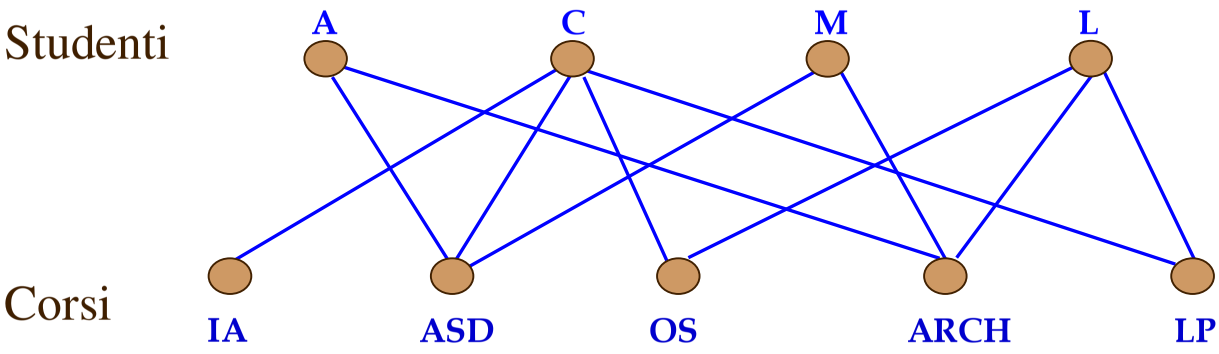
Prof. Sebastiano Battiato

Prof.ssa Rosalba Giugno

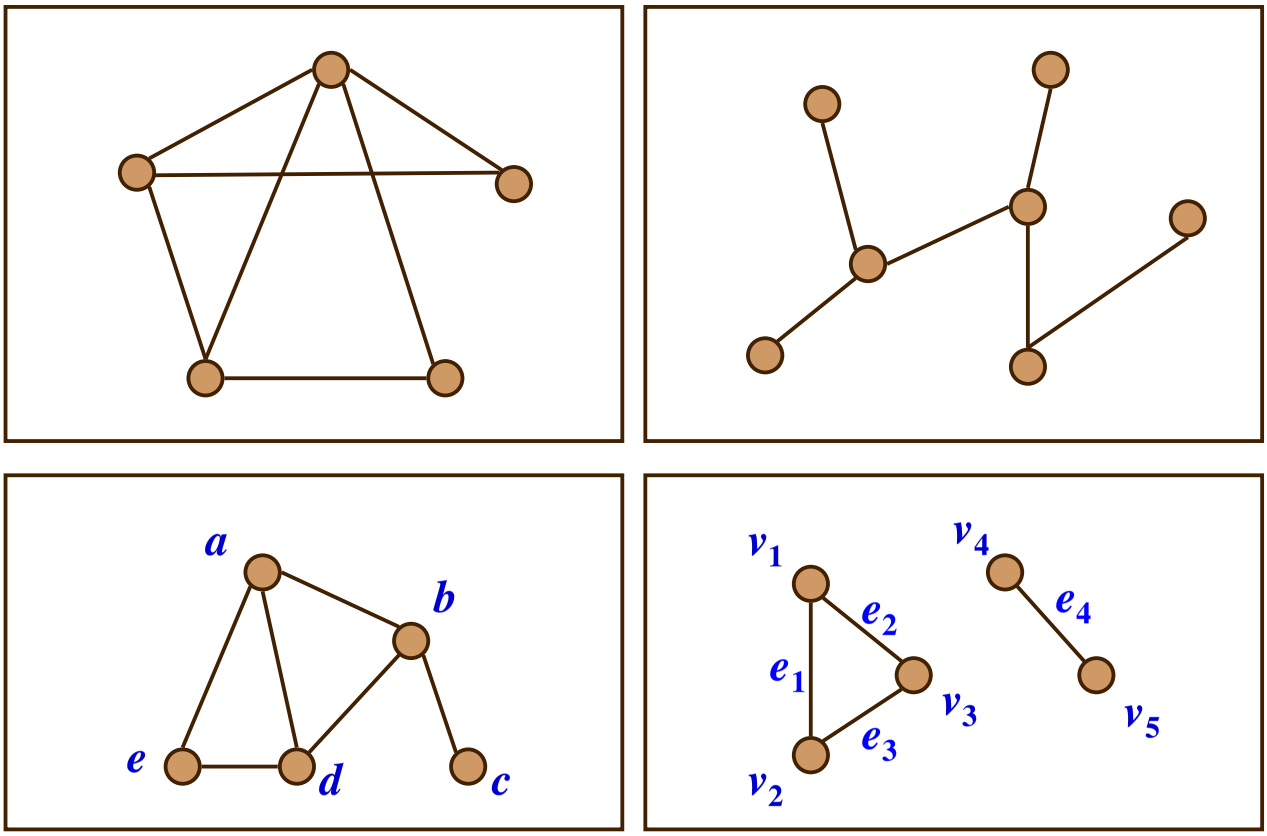
## Cos'è un grafo

Esempio:

| Studenti | Corsi |
|---|---|
| **Marco** | **ASD, ARCH** |
| **Carla** | **IA, ASD, OS, LP** |
| **Andrea** | **ASD, ARCH** |
| **Laura** | **OS, ARCH, LP** |



# Esempi di grafi




# Definizione di grafo

Un **grafo** $G$ è una coppia di elementi $(V, E)$ dove:

$V$ è un insieme detto insieme dei **vertici**

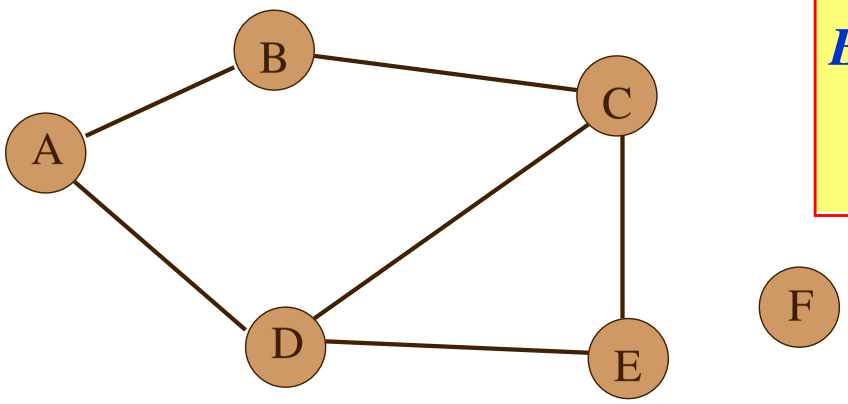
$E$ è un isieme detto insieme degli **archi**

# Definizione di grafo

Un **grafo** ***G*** è una coppia di elementi $(V, E)$ dove:

$V$ è un insieme detto insieme dei **vertici**

$E$ è un isieme detto insieme degli **archi**

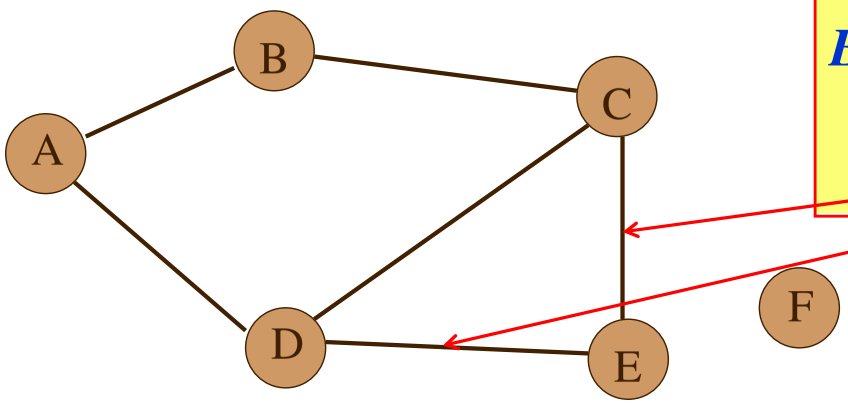



$V = \{A, B, C, D, E, F\}$

$E = \{(A,B), (A,D), (B,C), (C,D), (C,E), (D,E)\}$


# Definizione di grafo


Un **arco** è una coppia di vertici $(v,w)$, cioè





$V = \{A, B, C, D, E, F\}$

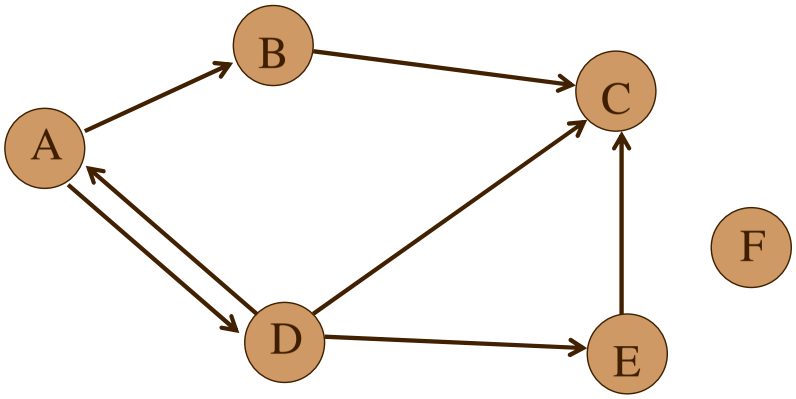
$E = \{(A,B), (A,D), (B,C), (C,D), (C,E), (D,E)\}$

# Tipi di grafi: grafi orientati

Un **grafo orientato *G*** è una coppia **(*V*, *E*)** dove:

***V*** è un insieme detto insieme dei **vertici**

***E*** è una ***relazione binaria*** tra vertici detta insieme degli **archi**




# Tipi di grafi: grafi orientati

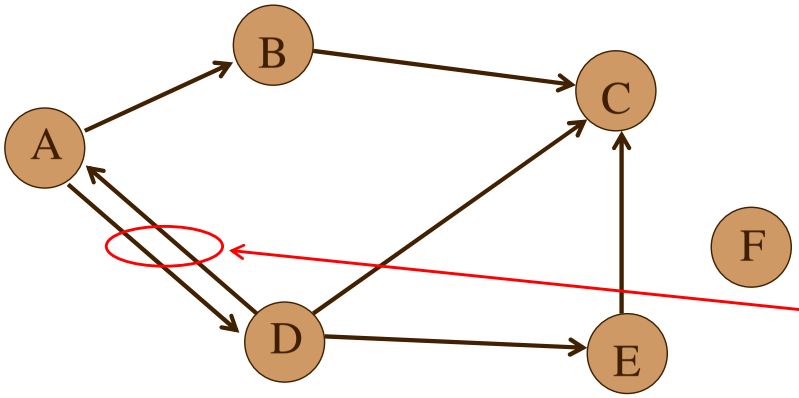
Un **grafo orientato *G*** è una coppia **(*V*, *E*)** dove:

***V*** è un insieme detto insieme dei **vertici**

***E*** è una ***relazione binaria*** tra vertici detta insieme degli **archi**

**$V$ = {A, B, C, D, E, F}**

**$E$ ={(A,B), (A,D), (B,C), (D,C), (E,C),(D,E), (D,A)}**



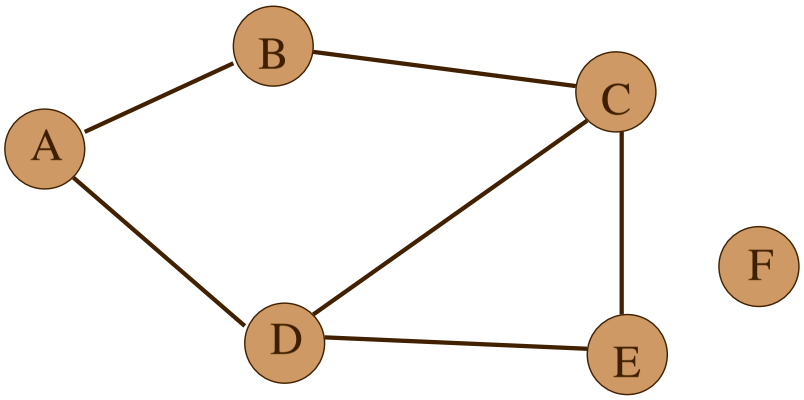

**(A,D) *e* (D,A) denotano due archi diversi**

# Tipi di grafi: grafi non orientati

Un **grafo non orientato *G*** è una coppia **(*V*, *E*)** dove:

***V*** è un insieme detto insieme dei **vertici**

***E*** è un isieme ***non ordinato*** di coppie di vertici detto insieme degli **archi**



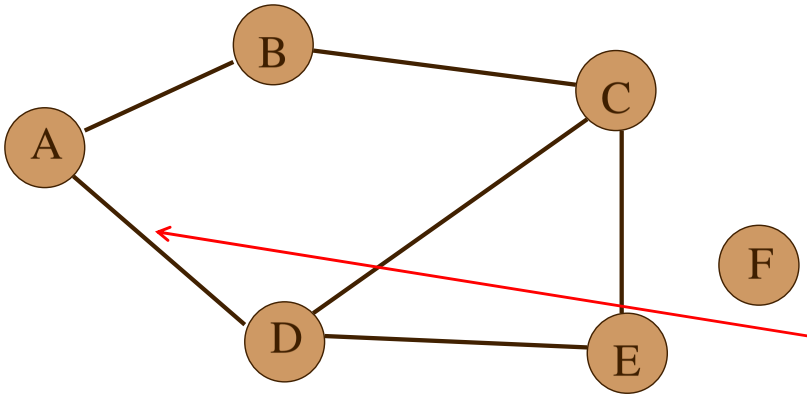

**$V$ = {A, B, C, D, E, F}**
**$E$ = {(A,B), (A,D), (B,C), (C,D), (C,E), (D,E)}**


# Tipi di grafi: grafi non orientati

Un **grafo non orientato *G*** è una coppia **(*V*, *E*)** dove:

***V*** è un insieme detto insieme dei **vertici**

***E*** è un isieme ***non ordinato*** di coppie di vertici detto insieme degli **archi**

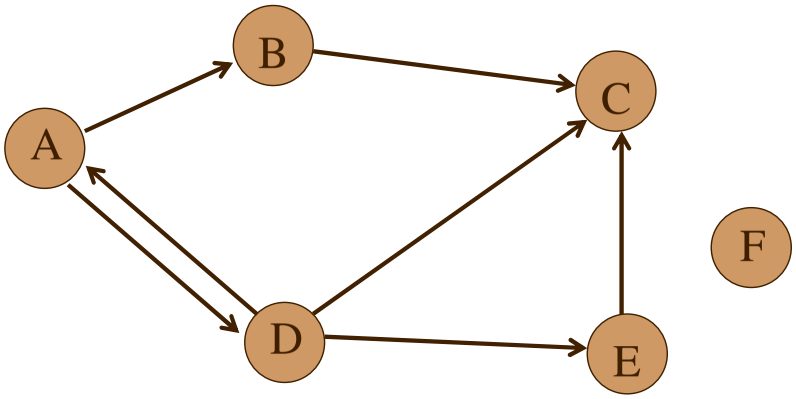


**$V$ = {A, B, C, D, E, F}**
**$E$ = {(A,B), (A,D), (B,C), (C,D), (C,E), (D,E)}**


**(A,D) *e* (D,A) denotano lo stesso arco**

# Definizioni sui grafi

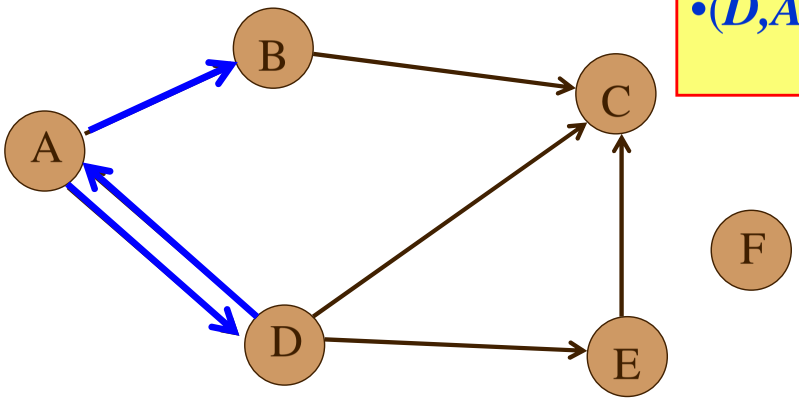
In un grafo orientato, un arco ***(w,v)*** si dice ***incidente*** da ***w*** in ***v***




# Definizioni sui grafi

In un grafo orientato, un arco ***(w,v)*** si dice ***incidente*** da ***w*** in ***v***

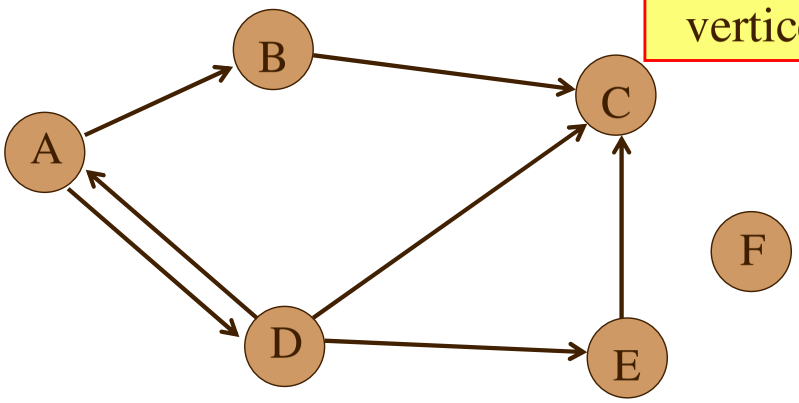
- ***(A,B)*** è ***incidente*** da ***A*** a ***B***
- ***(A,D)*** è ***incidente*** da ***A*** a ***D***
- ***(D,A)*** è ***incidente*** da ***D*** a ***A***



Un vertice *w* si dice ***adiacente*** a *v* se e solo se

- **(*v*, *w*) appartiene a**

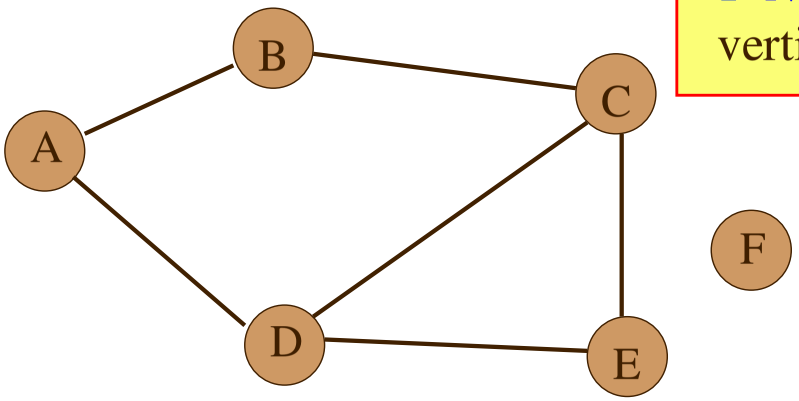
- ***B*** è ***adiacente*** ad ***A***
- ***C*** è ***adiacente*** a ***B*** e a ***D***
- ***A*** è ***adiacente*** a ***D*** e vice versa
- ***B NON*** è ***adiacente*** a ***D NÉ*** a ***C***
- ***F NON*** è ***adiacente*** ad alcun vertice




Definizioni sui grafi

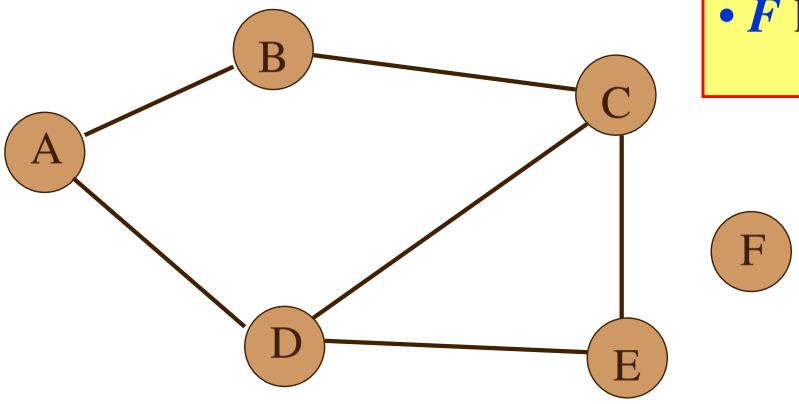
In un ***grafo non orientato*** la relazione di ***adiacenza*** tra vertici è ***simmetrica***

- ***A*** è ***adiacente*** a ***D*** e vice versa
- ***B*** è ***adiacente*** a ***A*** e vice versa
- ***F NON*** è ***adiacente*** ad alcun vertice



# Definizioni sui grafi

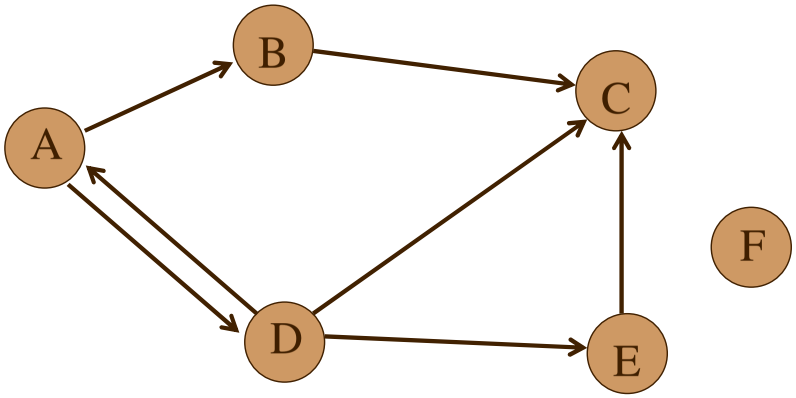
In un ***grafo non orientato*** il ***grado*** di un ***vertice*** è il ***numero di archi*** che da esso si dipartono




- ***A***, ***B*** ed ***E*** hanno ***grado 2***
- ***C*** e ***D*** hanno ***grado 3***
- ***F*** ha ***grado 0***

# Definizioni sui grafi

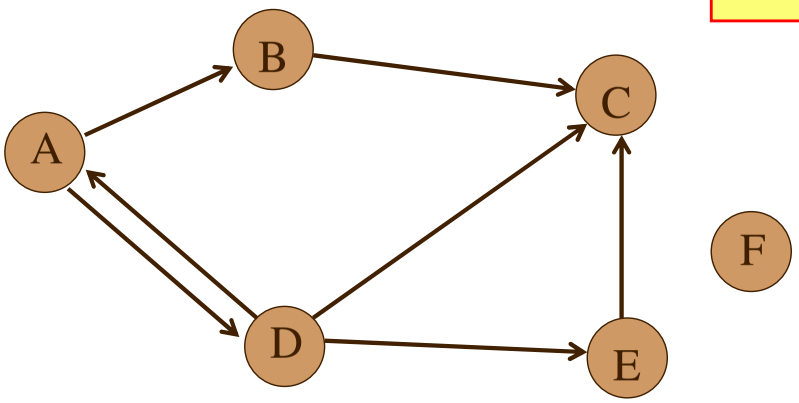
In un ***grafo orientato*** il ***grado entrante (uscente)*** di un ***vertice*** è il ***numero di archi incidenti*** in (da) esso




- ***A*** ha ***grado uscente 2*** e ***grado entrante 1***
- ***B*** ha ***grado uscente 1*** e ***grado entrante 1***
- ***C*** ha ***grado uscente 0*** e ***grado entrante 3***
- ***D*** ha ***grado uscente 3*** e ***grado entrante 1***

In un ***grafo orientato*** il ***grado*** di un ***vertice*** è la somma del suo ***grado entrante*** e del suo ***grado uscente***

- ***A*** e ***C*** hanno ***grado 3***
- ***B*** ha ***grado 2***
- ***D*** ha ***grado 4***

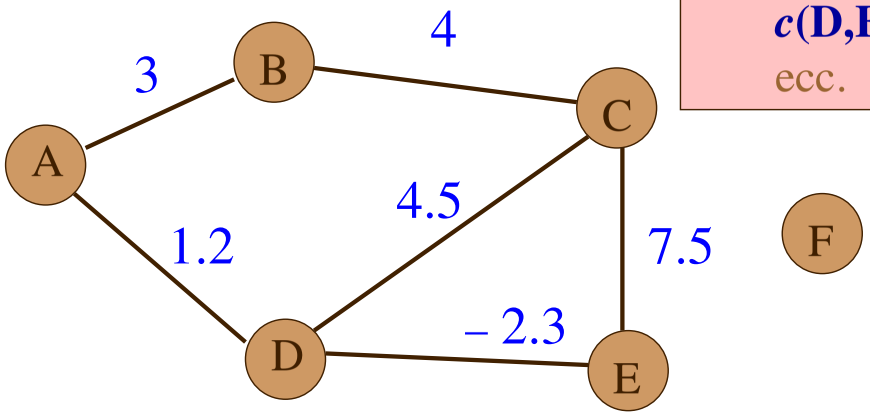



Definizioni sui grafi

In alcuni casi ogni arco ha un ***peso*** (o ***costo***) associato.

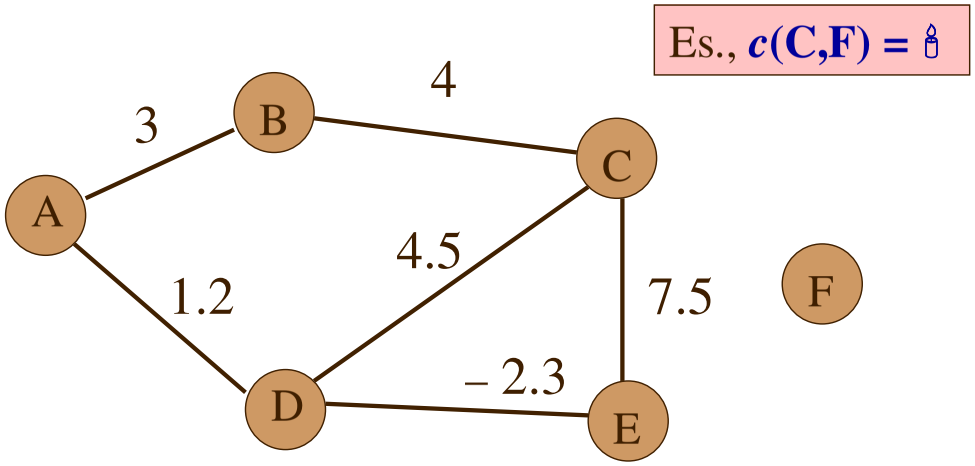
Il costo può essere determinato tramite una funzione di costo, $c: E \rightarrow \mathbf{R}$, dove **R** è l'insieme dei numeri reali (o interi).

Es., $c(A, B) = 3$, $c(D,E) = -2.3$, ecc.



In alcuni casi ogni arco ha un ***peso*** (o ***costo***) associato.

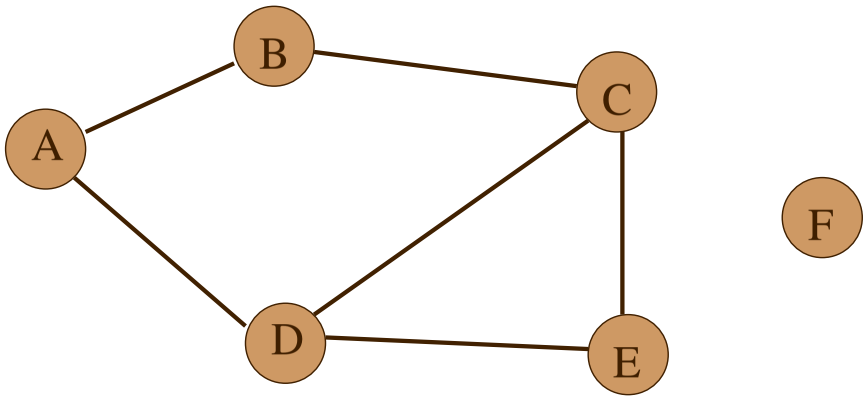
Quando tra due vertici ***non esiste*** un arco, si dice che il costo è ***infinito***.




Definizioni sui grafi

Sia $G = (V, E)$ un grafo.

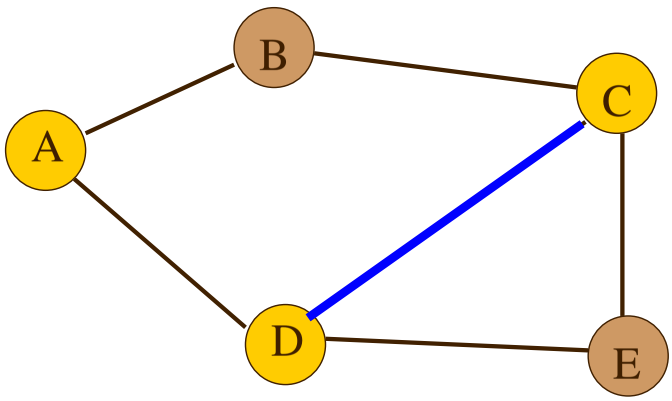
Un ***sottografo*** di $G$ è un grafo $H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$. (e poiché $H$ è un grafo, deve valere che $E^* \subseteq V^* \times V^*$.)



# Definizioni sui grafi

Sia $G = (V, E)$ un grafo.

Un ***sottografo*** di $G$ è un grafo $H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$. (e poiché $H$ è un grafo, deve valere che $E^* \subseteq V^* \times V^*$.)




Es.,

$V^* = \{A, C, D\}$,

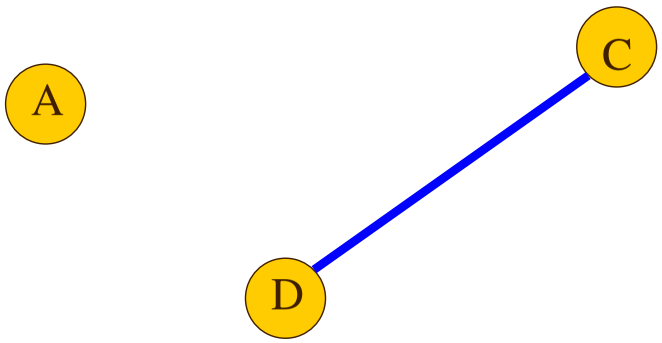
$E^* = \{(C, D)\}$.

F


# Definizioni sui grafi

Sia $G = (V, E)$ un grafo.

Un ***sottografo*** di $G$ è un grafo $H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$. (e poiché $H$ è un grafo, deve valere che $E^* \subseteq V^* \times V^*$.)



Es.,

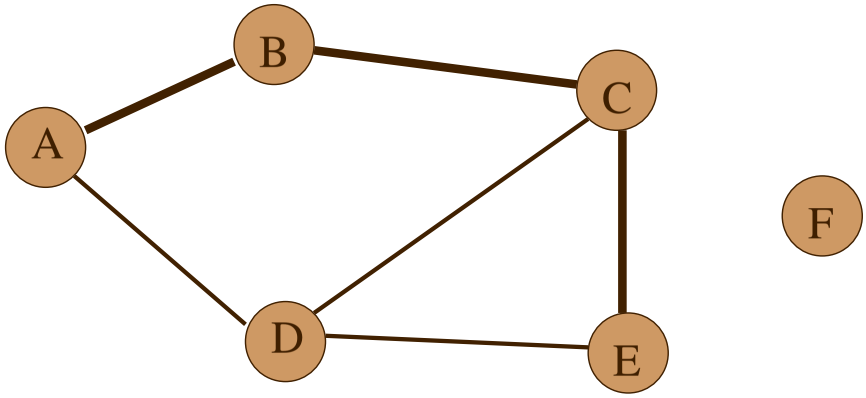
$V^* = \{A, C, D\}$,

$E^* = \{(C, D)\}$.

# Definizioni sui grafi

Sia $G = (V, E)$ un grafo.

Un ***percorso*** nel grafo è una sequenza di vertici $\langle w_1, w_2, ..., w_n \rangle$ tale che $(w_i, w_{i+1}) \in E$ per $1 \leq i \leq n-1$.

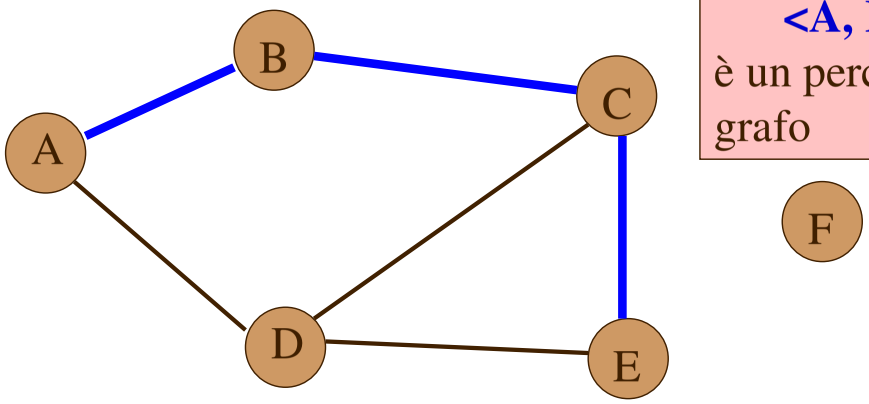




# Definizioni sui grafi

Sia $G = (V, E)$ un grafo.

Un ***percorso*** nel grafo è una sequenza di vertici $\langle w_1, w_2, ..., w_n \rangle$ tale che $(w_i, w_{i+1}) \in E$ per $1 \leq i \leq n-1$.

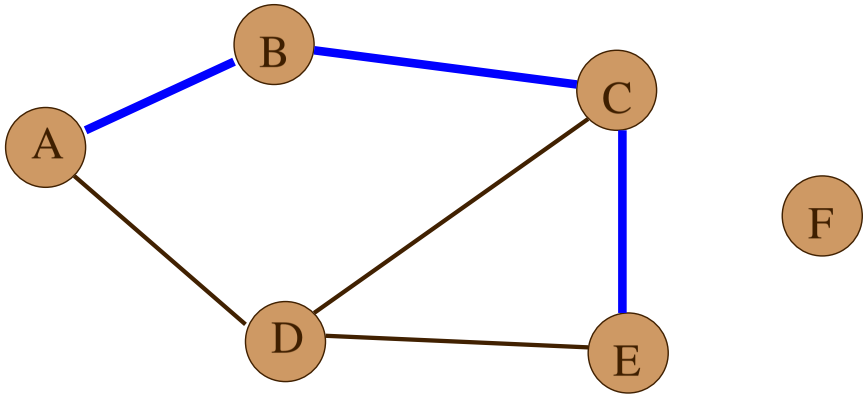




Es.,
**<A, B, C, E>**
è un percorso nel grafo

Sia $G = (V, E)$ un grafo.

Un ***percorso*** nel grafo è una sequenza di vertici $\langle w_1, w_2, ..., w_n \rangle$ tale che $(w_i, w_{i+1}) \in E$ per $1 \leq i \leq n-1$.

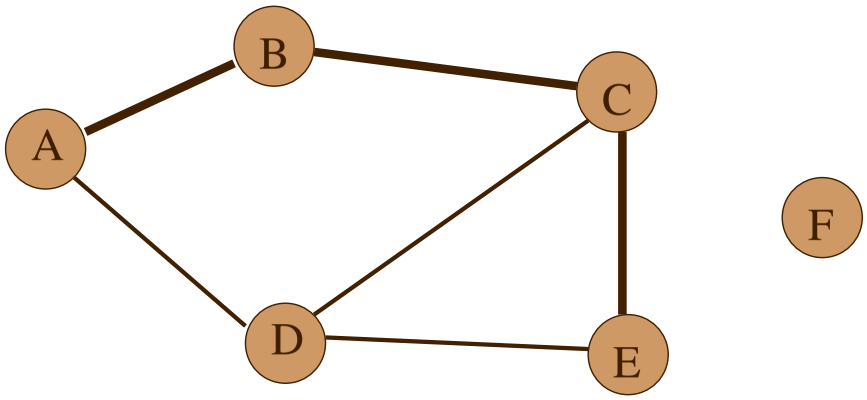
Il ***percorso*** $\langle w_1, w_2, ..., w_n \rangle$ si dice che ***contiene*** i vertici $w_1, w_2, \ldots, w_n$ e gli archi $(w_1,w_2)\ (w_2,w_3)\ ...(w_{n-1},w_n)$




Definizioni sui grafi

Sia $\langle w_1, w_2, \ldots, w_n \rangle$ un ***percorso***.

La ***lunghezza*** del percorso è il ***numero totale di archi*** che connettono i vertici nell'ordine della sequenza ($n-1$, uno in meno del numero di vertici nella sequenza).

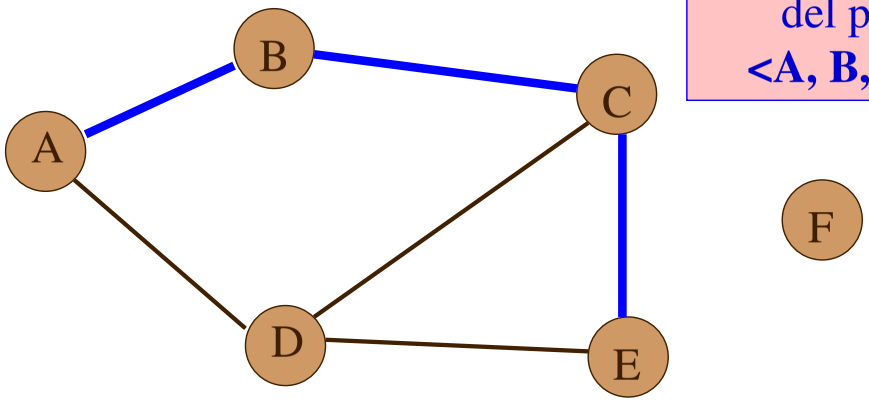


# Definizioni sui grafi

Sia $\langle w_1, w_2, \ldots, w_n \rangle$ un ***percorso***.

La ***lunghezza*** del percorso è il ***numero totale di archi*** che connettono i vertici nell'ordine della sequenza ($n$–1, uno in meno del numero di vertici nella sequenza).

Es., la unghezza del percorso **<A, B, C, E>** è **3**.

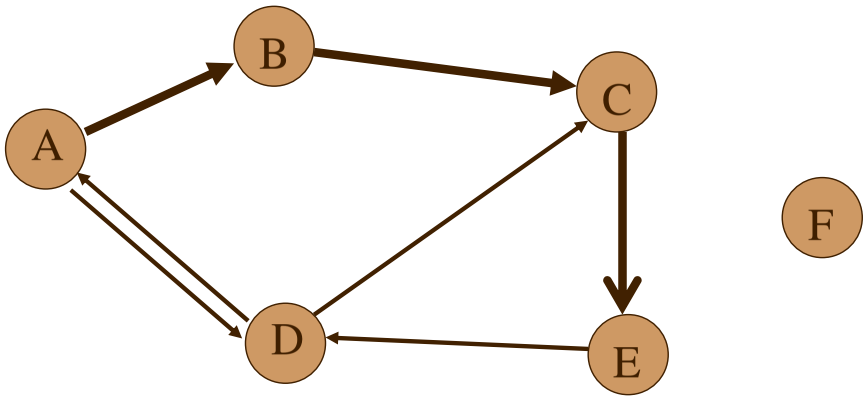




# Definizioni sui grafi


Sia $\langle w_1, w_2, \ldots, w_n \rangle$ un ***percorso*** in un ***grafo orientato***.

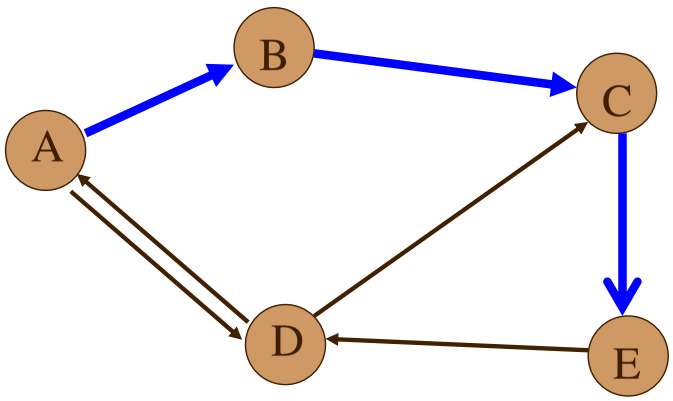
Poiché ***ogni arco*** $(w_i, w_{i+1})$ nel percorso è ***ordinato***, gli ***archi*** del percorso sono sempre ***orientati lungo il percorso***.



# Definizioni sui grafi

Sia $\langle w_1, w_2, \ldots, w_n \rangle$ un ***percorso*** in un ***grafo orientato***.

Poiché ***ogni arco*** $(w_i, w_{i+1})$ nel percorso è ***ordinato***, gli ***archi*** del percorso sono sempre ***orientati lungo il percorso***.



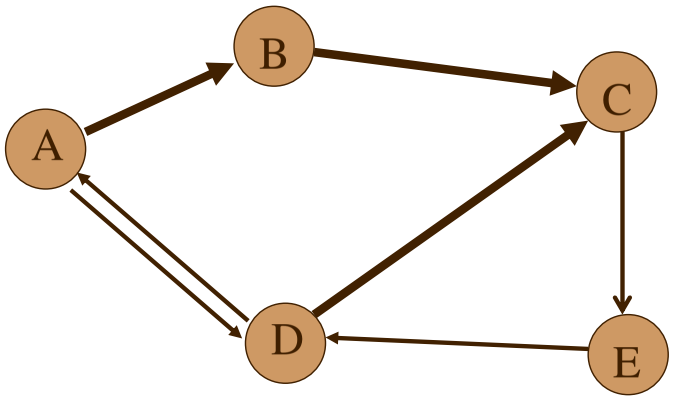

Es., <A, B, C, E> è un percorso in questo grafo orientato

F


# Definizioni sui grafi

Sia $\langle w_1, w_2, \ldots, w_n \rangle$ un ***percorso*** in un ***grafo orientato***.

Poiché ***ogni arco*** $(w_i, w_{i+1})$ nel percorso è ***ordinato***, gli ***archi*** del percorso sono sempre ***orientati lungo il percorso***.

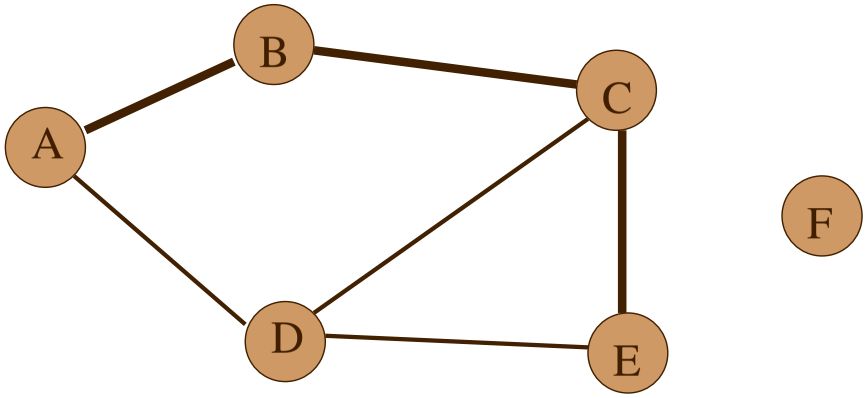




... ma **<A, B, C, D>** ***non è*** un percorso, poiché **(C, D)** non è un arco.

F

# Definizioni sui grafi

Un percorso si dice ***semplice*** se tutti i ***suoi vertici sono distinti*** (compaiono una sola volta nella sequenza), eccetto al più il primo e l'ultimo che possono essere lo stesso.

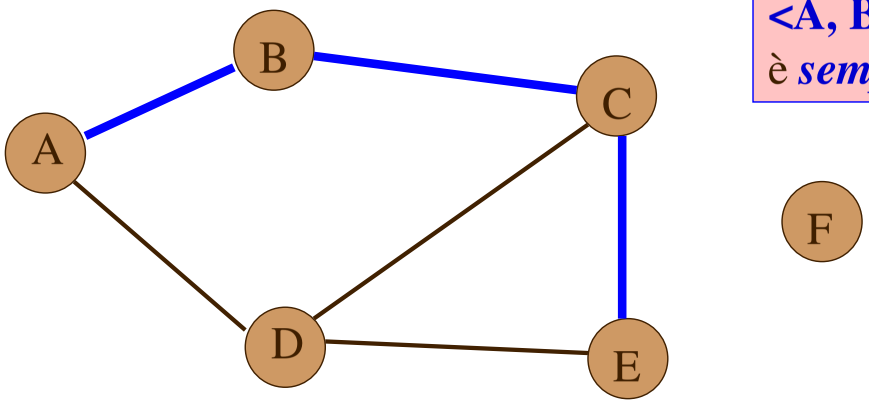




# Definizioni sui grafi

Un percorso si dice ***semplice*** se tutti i ***suoi vertici sono distinti*** (compaiono una sola volta nella sequenza), eccetto al più il primo e l'ultimo che possono esser lo stesso.

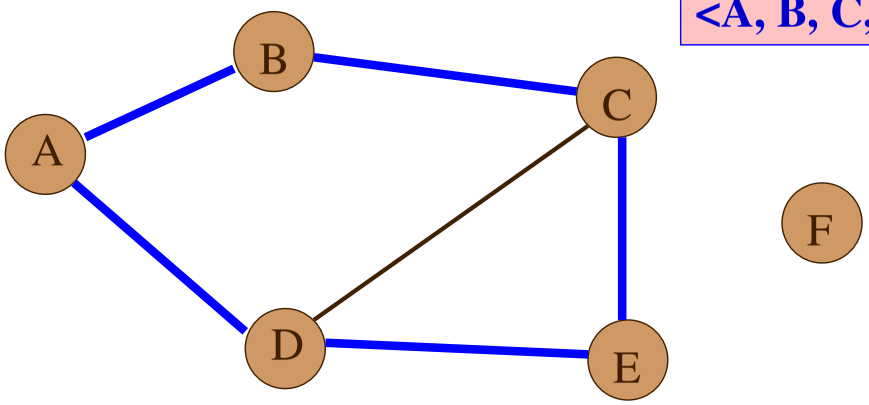

Es., il percorso **<A, B, C, E>** è ***semplice*** ...



# Definizioni sui grafi

Un percorso si dice ***semplice*** se tutti i ***suoi vertici sono distinti*** (compaiono una sola volta nella sequenza), eccetto al più il primo e l'ultimo che possono esser lo stesso.

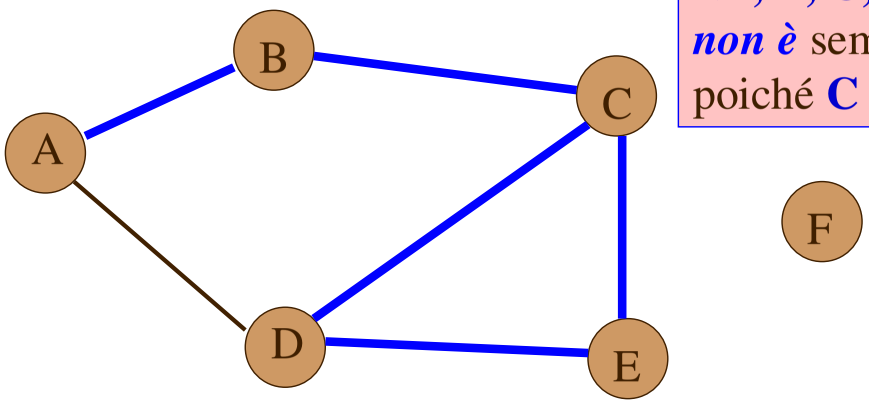
... così come anche **<A, B, C, E, D, A>**.




# Definizioni sui grafi

Un percorso si dice ***semplice*** se tutti i ***suoi vertici sono distinti*** (compaiono una sola volta nella sequenza), eccetto al più il primo e l'ultimo che possono esser lo stesso.

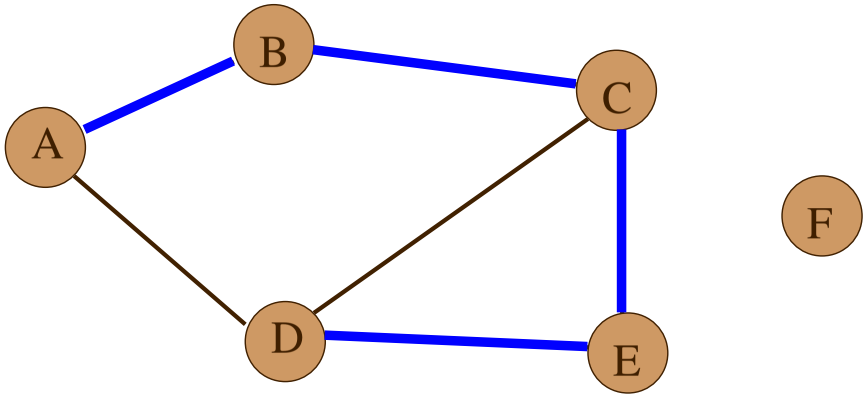
... ma il percorso **<A, B, C, E, D, C >** ***non è*** semplice, poiché **C** è ripetuto.



# Definizioni sui grafi

Se esiste un percorso ***p*** tra i vertici ***v*** e ***w***, si dice che ***w è raggiungibile da v*** tramite ***p*** $v \xrightarrow[p]{} w$

Es.: ***A*** è ***raggiungibile*** da ***D*** e vice versa

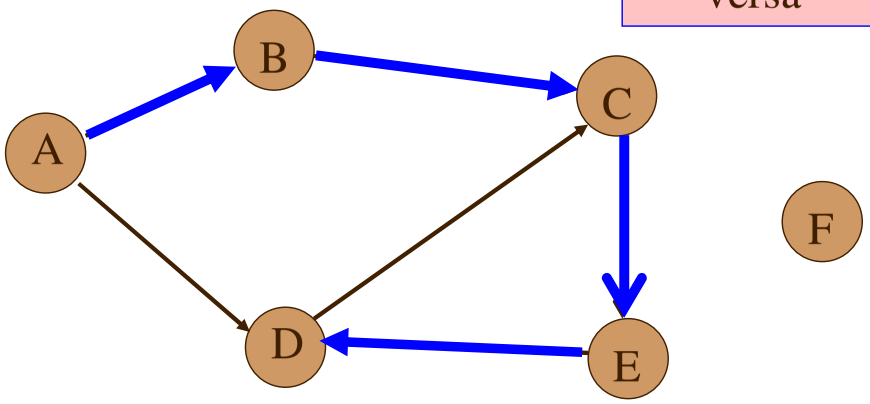




# Definizioni sui grafi

Se esiste un percorso ***p*** tra i vertici ***v*** e ***w***, si dice che ***w è raggiungibile da v*** tramite ***p*** $v \xrightarrow[p]{} w$


Es.: ***D*** è ***raggiungibile*** da ***A*** ma non vice versa

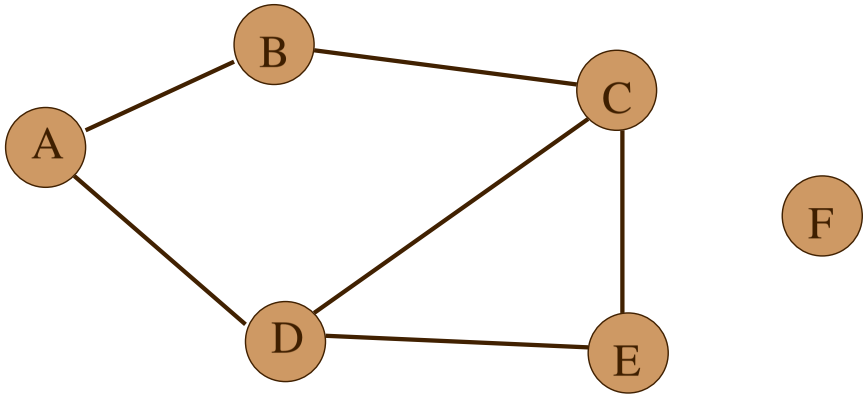


# Definizioni sui grafi

Se ***G*** è un ***grafo non orientato***, diciamo che ***G*** è ***connesso*** se esiste un ***percorso*** da ***ogni vertice*** ad ***ogni altro vertice***.

**Un grafo non orientato *non connesso* si dice *sconnesso*.**

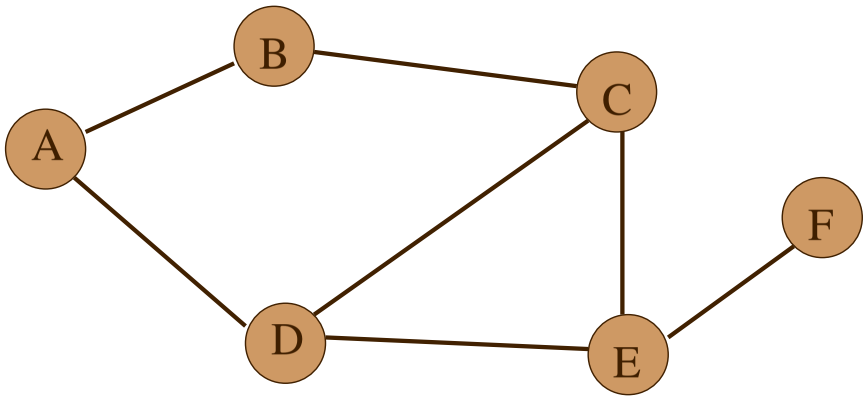
**Questo grafo non orientato *non è connesso*.**




# Definizioni sui grafi

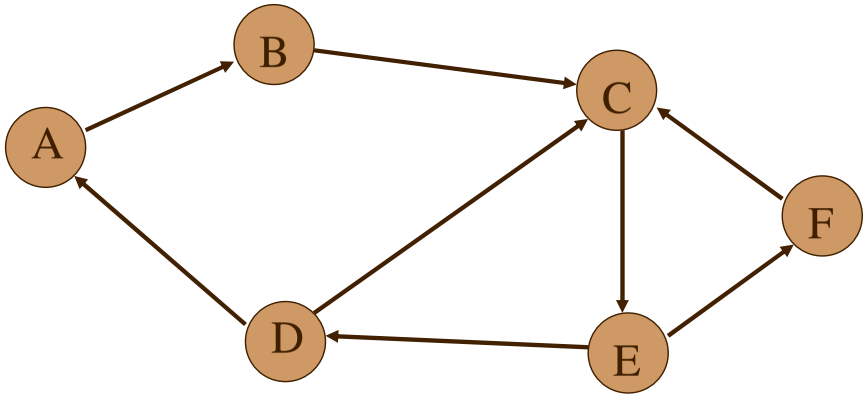
Se ***G*** è un ***grafo non orientato***, diciamo che ***G*** è ***connesso*** se esiste un ***percorso*** da ***ogni vertice*** ad ***ogni altro vertice***.

Questo ***è connesso***.



Se ***G*** è un ***grafo orientato***, diciamo che ***G*** è ***fortemente connesso*** se esiste un ***percorso*** da ***ogni vertice*** ad ***ogni altro vertice***.

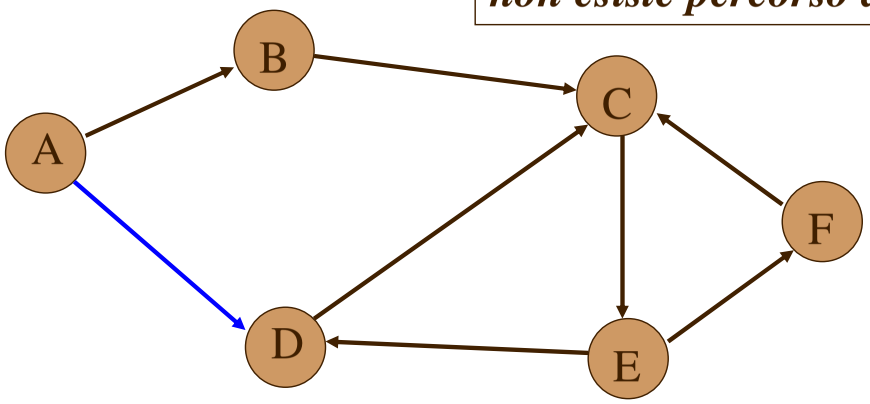
Questo grafo orientato ***è fortemente connesso.***




Definizioni sui grafi

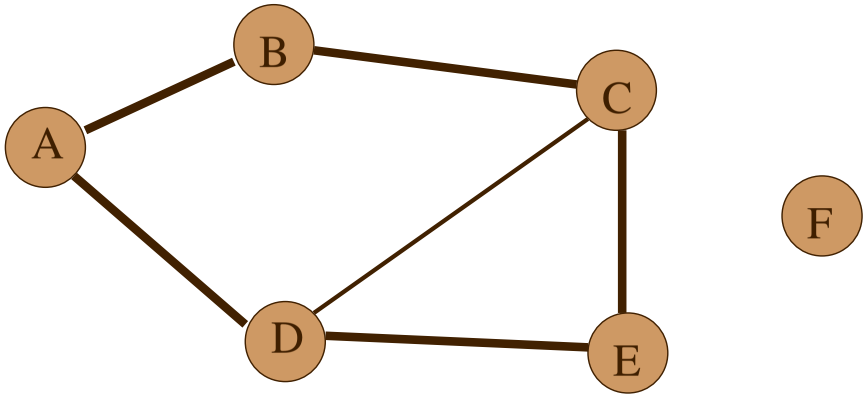
Se ***G*** è un ***grafo orientato***, diciamo che ***G*** è ***fortemente connesso*** se esiste un ***percorso*** da ***ogni vertice*** ad ***ogni altro vertice***.

Questo grafo orientato **non è *fortemente connesso*; ad es., *non esiste percorso da* D *a* A**.



# Definizioni sui grafi

Un ***ciclo*** in un grafo è un percorso $\langle w_1, w_2, \ldots, w_n \rangle$ di lunghezza almeno 1, tale che $w_1 = w_n$.

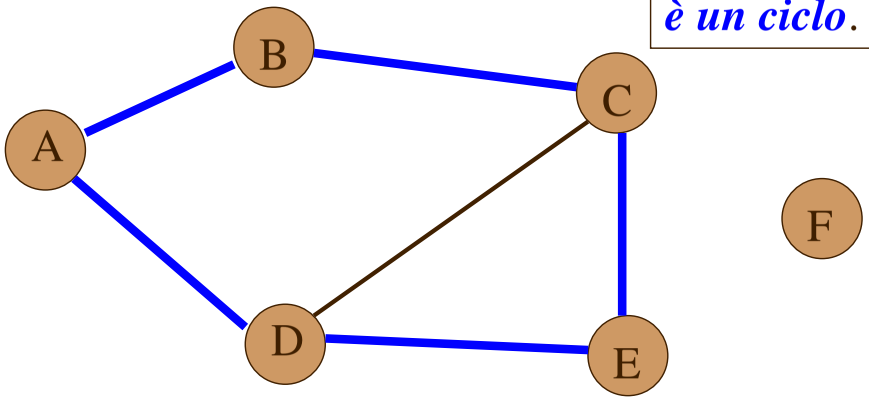




# Definizioni sui grafi

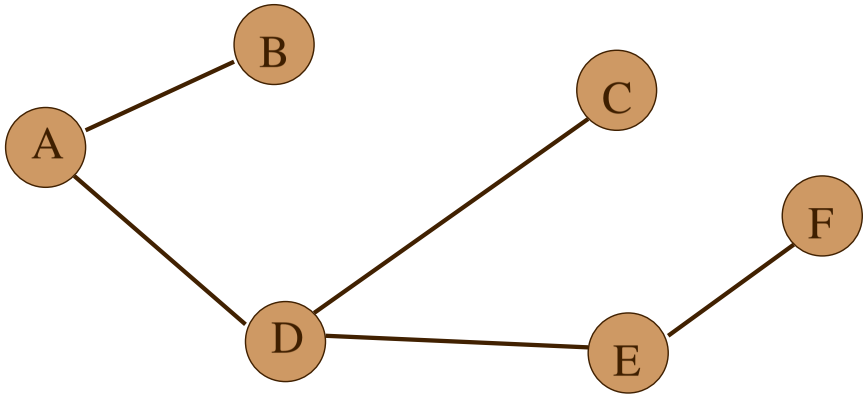
Un ***ciclo*** in un grafo è un percorso $\langle w_1, w_2, \ldots, w_n \rangle$ di lunghezza almeno 1, tale che $w_1 = w_n$.


Es.: il percorso **<A, B, C, E, D, A>** ***è un ciclo***.



Un ***grafo senza cicli*** è detto ***aciclico***.

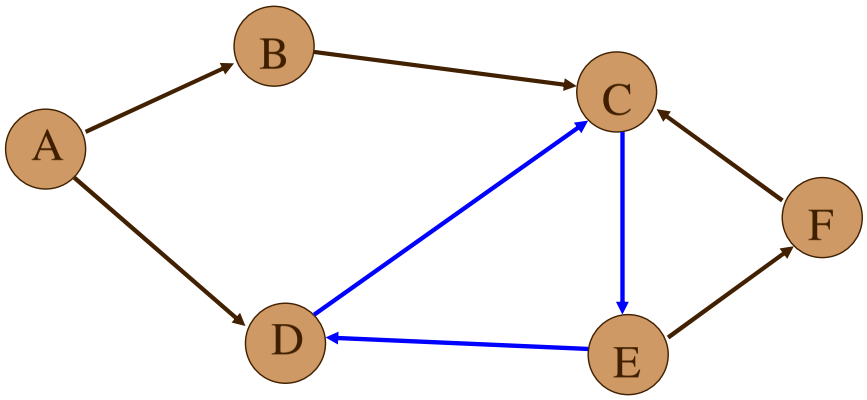
Questo grafo ***è aciclico***.




Definizioni sui grafi

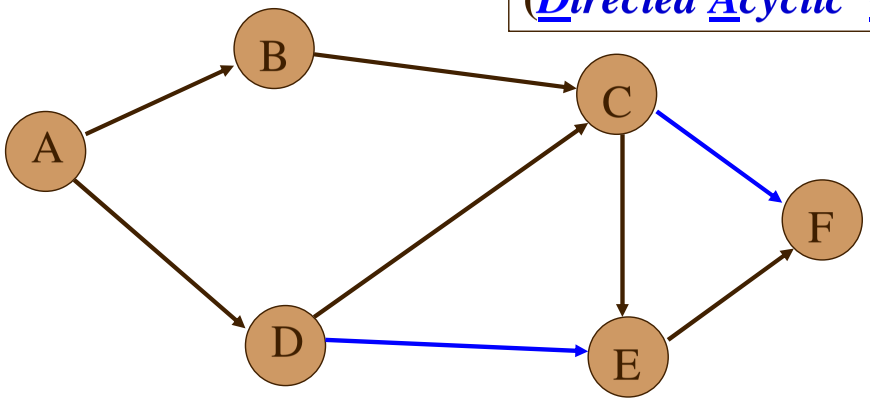
Un ***grafo senza cicli*** è detto ***aciclico***.

Questo grafo orientato ***non è aciclico***, ...



Un ***grafo senza cicli*** è detto ***aciclico***.

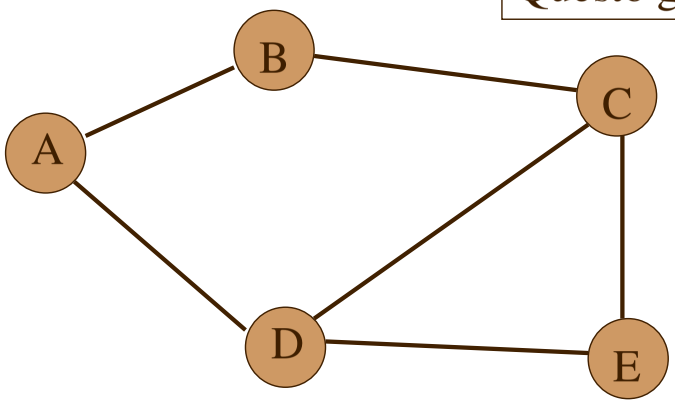
... ma questo lo è.
Un ***grafo orientato aciclico*** è spesso chiamato **DAG** (***Directed Acyclic Graph***).




Definizioni sui grafi

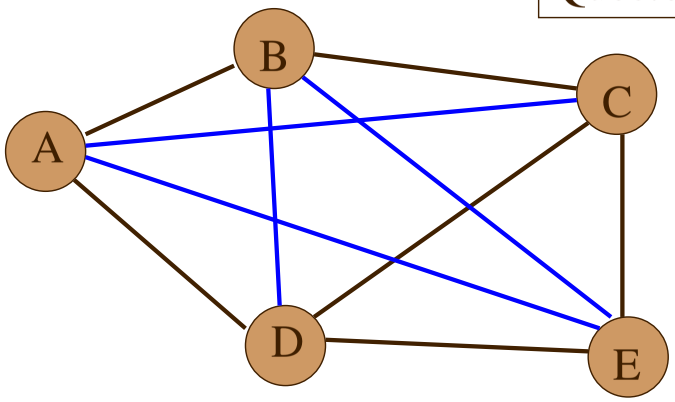
Un ***grafo completo*** è un grafo che ha un ***arco tra ogni coppia di vertici***.

Questo grafo ***non è completo***



Un ***grafo completo*** è un grafo che ha un ***arco tra ogni coppia di vertici***.

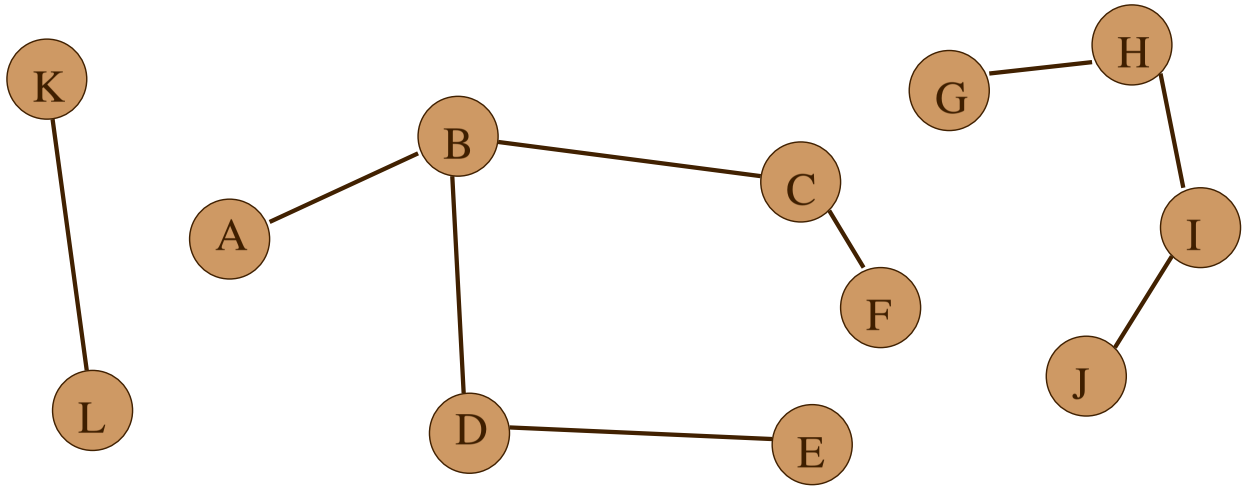
Questo grafo ***è completo***




Definizioni sui grafi

Se un ***grafo non orientato*** è ***aciclico*** ma ***sconnesso***, prende il nome di ***foresta***.

Questa è una ***foresta***. Contiene tre alberi liberi.



**Ci sono due tipi *standard* di rappresentazioni di grafi in un computer:**

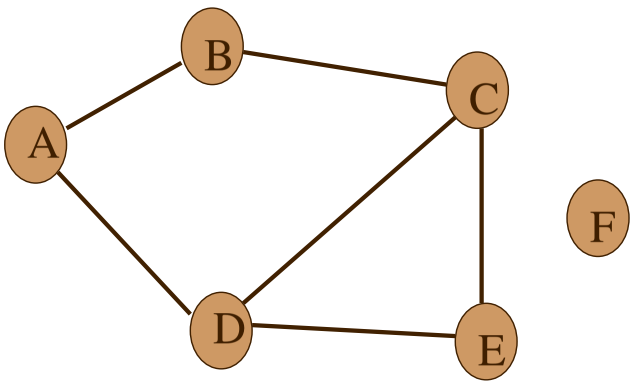
- **Rappresentazione con *matrice di adiacenza***
- **Rappresentazione con *liste di adiacenza***

Rappresentazone di grafi

Rappresentazione con ***matrice di adiacenza***:

$$M(v, w) = \begin{cases} \mathbf{1} & \text{se } (v, w) \in E \\ \mathbf{0} & \text{altrimenti} \end{cases}$$

**Spazio: $|V|^2$**

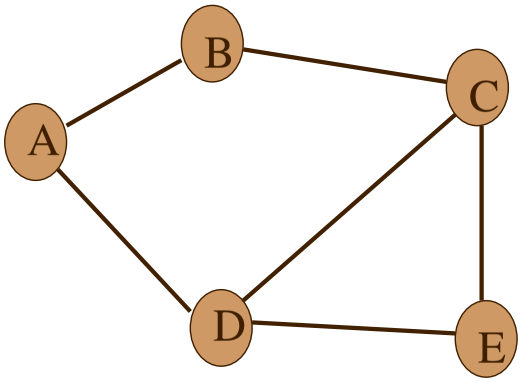



| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | 1 | 0 | 1 | 0 | 0 |
| B | 1 | 0 | 1 | 0 | 0 | 0 |
| C | 0 | 1 | 0 | 1 | 1 | 0 |
| D | 1 | 0 | 1 | 0 | 1 | 0 |
| E | 0 | 0 | 1 | 1 | 0 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 0 |

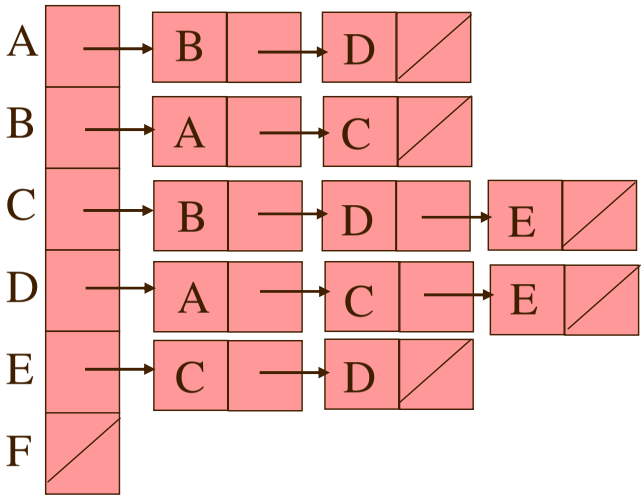
# Rappresentazone di grafi

Rappresentazione con ***liste di adiacenza***:

$L(v)$ = lista di $w$, tale che $(v, w) \in E$,
per $v \in V$




F



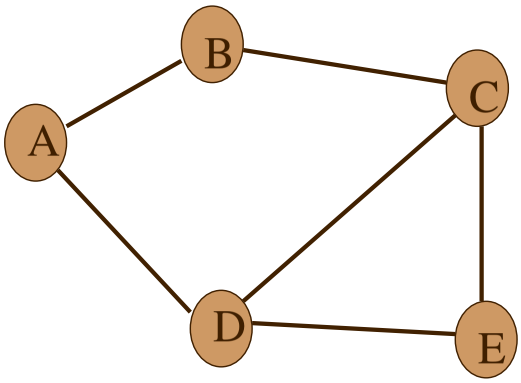


# Rappresentazone di grafi

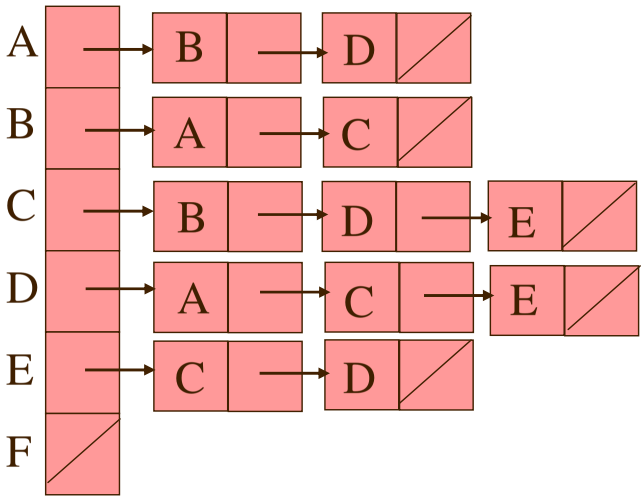
Rappresentazione con ***liste di adiacenza***:

$L(v)$ = lista di $w$, tale che $(v, w) \in E$,
per $v \in V$


**Quanto spazio ?**




F

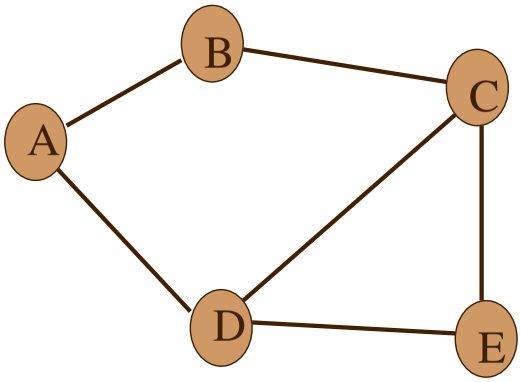


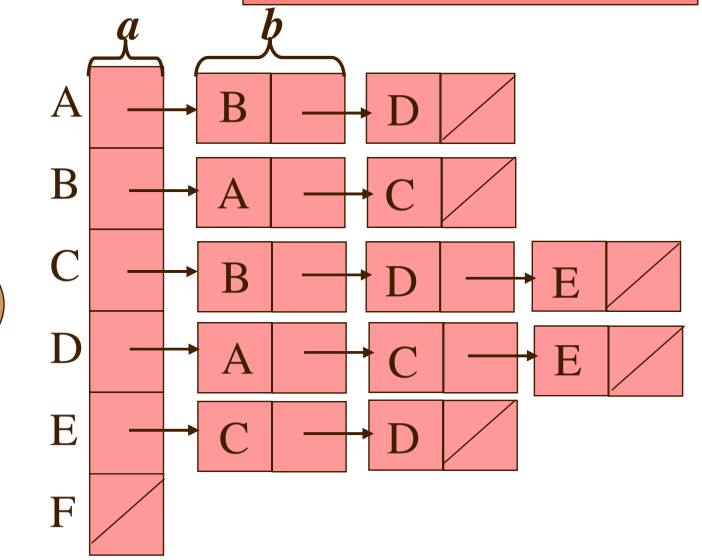
# Rappresentazone di grafi

Rappresentazione con ***liste di adiacenza***:

$L(v)$ = lista di $w$, tale che $(v, w) \in E$, per $v \in V$

**Spazio: $a\,|V| + 2\,b\,|E|$**

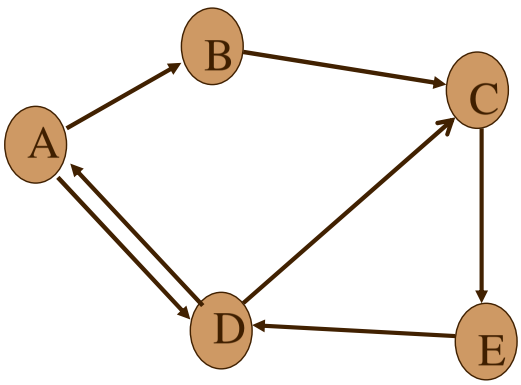




# Rappresentazone di grafi

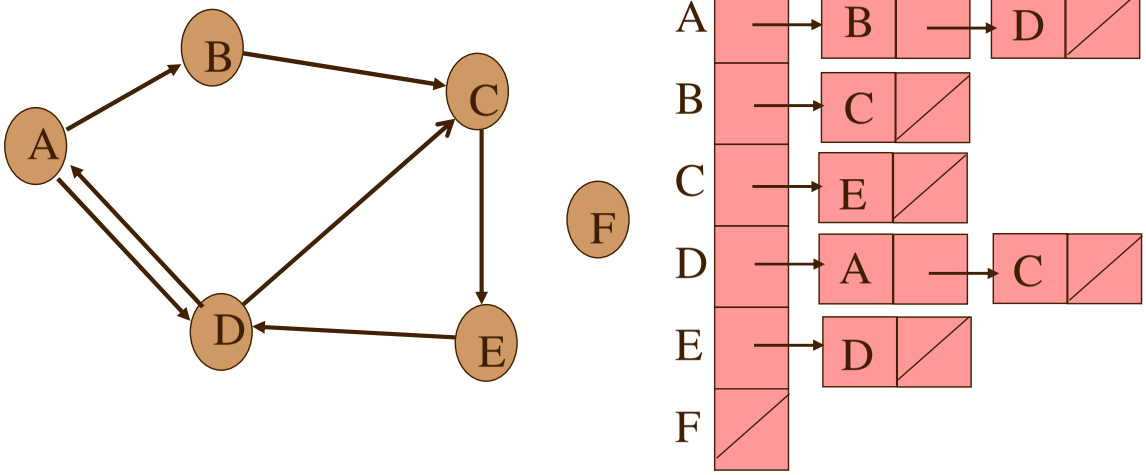

Rappresentazione con ***matrice di adiacenza*** questa volta per rappresentare un ***grafo orientato***.




| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | 1 | 0 | 1 | 0 | 0 |
| B | 0 | 0 | 1 | 0 | 0 | 0 |
| C | 0 | 0 | 0 | 0 | 1 | 0 |
| D | 1 | 0 | 1 | 0 | 0 | 0 |
| E | 0 | 0 | 0 | 1 | 0 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 0 |

# Rappresentazone di grafi

Rappresentazione con ***liste di adiacenza*** questa volta per rappresentare un ***grafo orientato***.

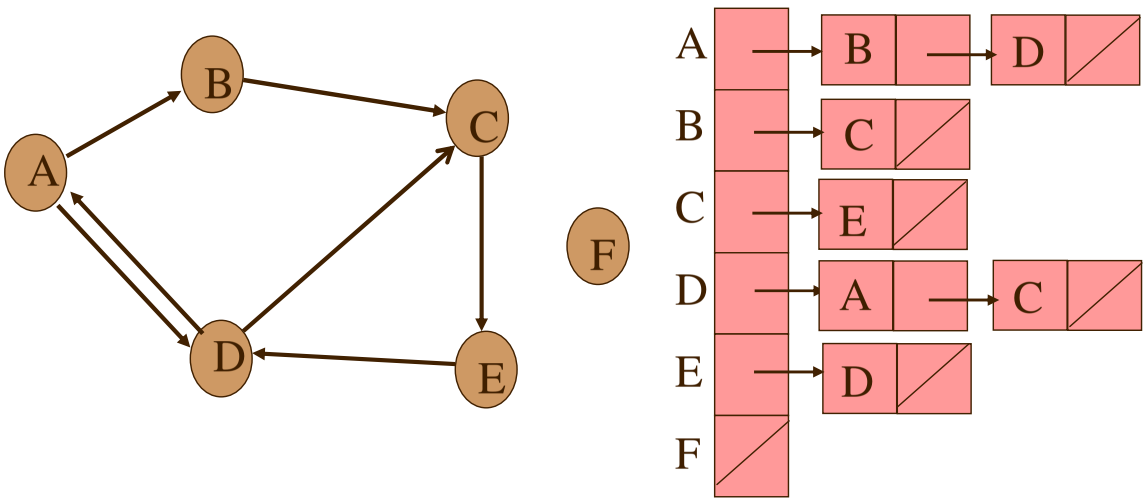




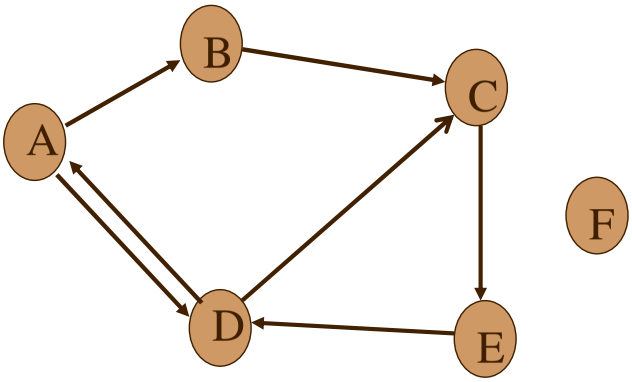
# Rappresentazone di grafi

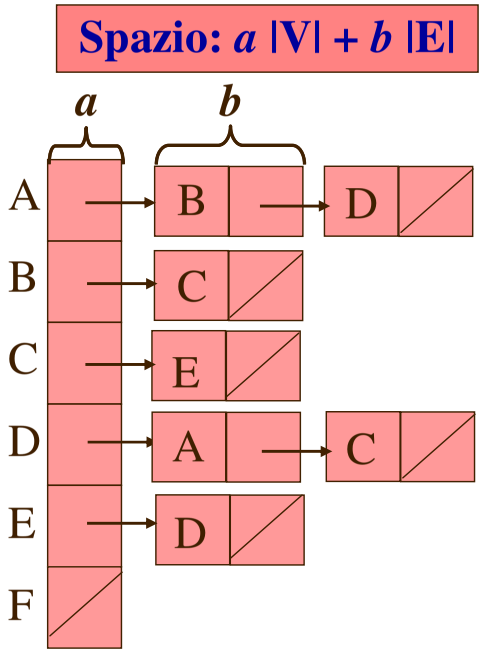
Rappresentazione con ***liste di adiacenza*** questa volta per rappresentare un ***grafo orientato***.


**Quanto spazio?**



Rappresentazione con ***liste di adiacenza*** questa volta per rappresentare un ***grafo orientato***.




Rappresentazone di grafi

## Matrice di adiacenza

- Spazio richiesto $O(|V|^2)$
- Verificare se i vertici *u* e *v* sono adiacenti richiede tempo O(1).
- Molti 0 nel caso di *grafi sparsi*

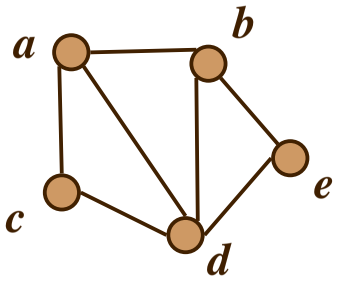
## Liste di adiacenza

- Spazio richiesto O(|E|+|V|)
- Verificare se i vertici *u* e *v* sono adiacenti richiede tempo O(|V|).

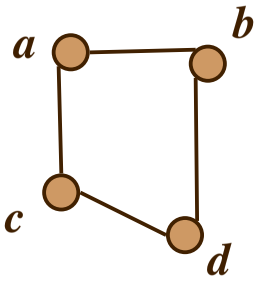
# Sottografo di copertura

Un *sottografo* di $G=(V,E)$ è un grafo $H = (V^*, E^*)$ tale che $V^* \subseteq V$ e $E^* \subseteq E$.

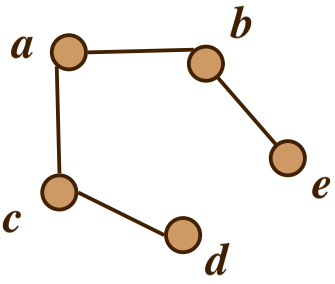
- ***H'* è un *sottografo di copertura* (o *di supporto* o sottografo "*spanning*") di *G* se**
  - $\boxed{V^* = V}$ **e** $E^* \subseteq E$




**G**




**H**



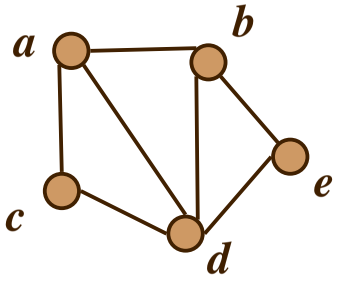

**H'**

# Albero di copertura

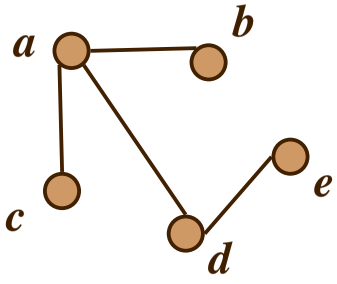
Un grafo $H=(V^*, E^*)$ è un *albero di copertura* (o *albero "spanning"*) *del grafo* $G=(V,E)$ se

*H* è un *grafo di copertura* di *G*

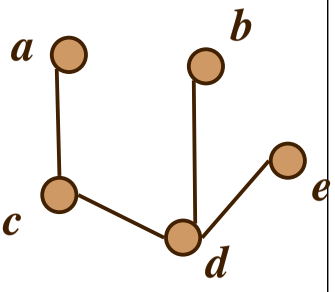
*H* è un *albero*




**G**




**H**




**H'**

# Visita in Profondità (DFS)

Tecnica di visita di un grafo

- È una variazione della visita in preordine per alberi binari

Si parte da un veritice *s*

- Viene processato il vertice *s*
- Ricorsivamente si processano tutti i vertici adiacenti ad *s*

Bisogna evitare di riprocessare vertici già visitati

- Bisogna evitare i cicli
- Nuovamente, quando un vertice è stato visitato e (poi) processato viene marcato opportunamente (*colorandolo*)

# DFS: intuizioni

I passi dell'algoritmo *DFS*

si sceglie un vertice non visitato *s*

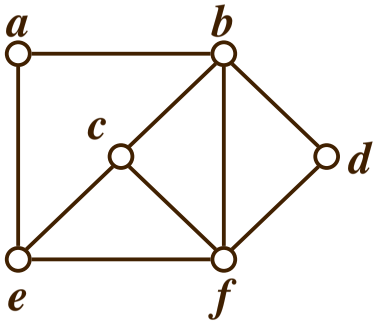
si sceglie un vertice non visitato adiacente ad *s*.

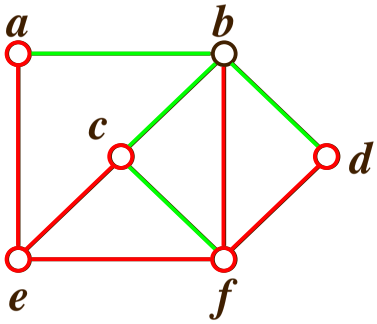
da *s* si attraversa quindi un percorso di vertici adiacenti (visitandoli) finché possibile:

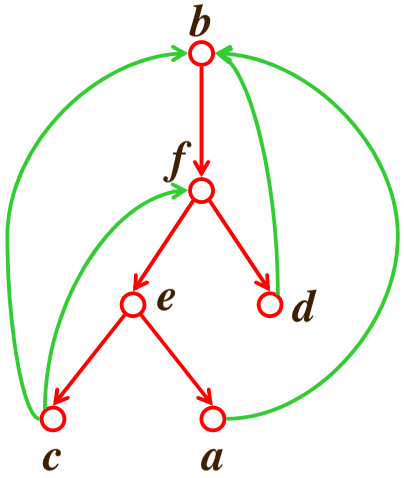
cioè finché non si incontra un vertice già visitato

appena si resta "bloccati" (tutti gli archi da un vertice sono stati visitati), si torna indietro (*backtracking*) di un passo (vertice) nel percorso visitato (aggiornando il vertice *s* al vertice corrente)

si ripete il processo ripartendo dal passo 🕐.




*b f e c a d*

**Albero di copertura Depth-first**

**Archi dell'albero** →
**Archi di ritoro** →



Manterremo traccia del *momento* (*tempo*) in cui ogni vertice $v$ viene *visitato (scoperto)* e del momento in cui viene *processato (terminato)*

Useremo due array $d[v]$ e $f[v]$ che registreranno il momento in cui $v$ verrà visitato e quello in cui verrà processato.

La variabile globale *tempo* serve a registrare il passaggio del tempo.

# Algoritmo DFS

```
DSF(G:grafo)
  for each vertice u ∈ V
       do  colore[u] = Binaco
           pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
       do if  colore[u] = Binaco
           then  DFS-Visita(u)
```

**Inizializzazione del grafo e della variabile tempo**

```
DSF-Visita(u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
       do if colore[v] = Binaco
           then pred[v] = u
                DFS-Visit(v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```

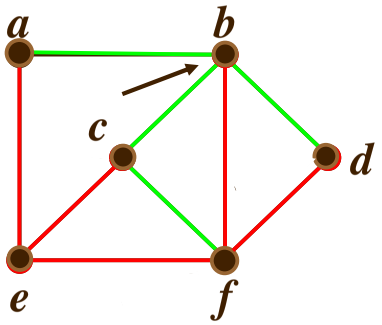
**Abbreviazione per:**
**tempo=tempo+1**
**d[u]=tempo**

**Abbreviazione per:**
**tempo=tempo+1**
**f[u]=tempo**

# DFS: simulazone

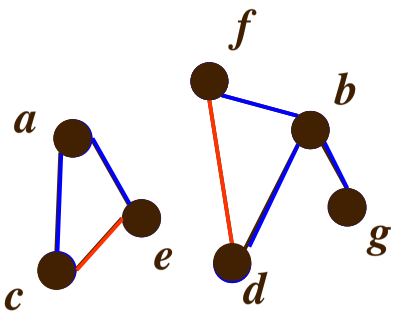
```
DSF(G:grafo)
  for each vertice u ∈ V
       do  colore[u] = Binaco
           pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
       do if  colore[u] = Binaco
          then  DFS-Visita(u)
```

```
DSF-Visita(u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
       do if colore[v] = Binaco
          then pred[v] = u
               DFS-Visit(v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```

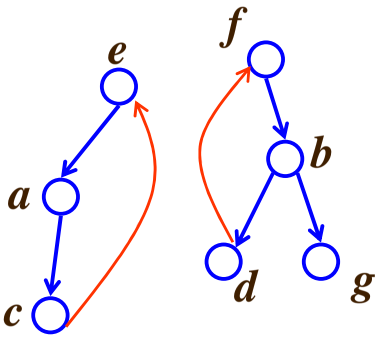


# Alberi di copertura multipli

```
DSF(G:grafo)
  for each vertice u ∈ V
       do  colore[u] = Binaco
           pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
       do if  colore[u] = Binaco
          then  DFS-Visita(u)
```




```
DSF-Visita(u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
       do if colore[v] = Binaco
          then pred[v] = u
               DFS-Visit(v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```




# Tempo di esecuzione di DSF

```
DSF(G:grafo)
  for each vertice u ∈ V
       do  colore[u] = Binaco
           pred[u] = NIL
  tempo = 0
  for each vertice u ∈ V
       do if  colore[u] = Binaco
          then  DFS-Visita(u)
```

$\Theta(|V|)$

$\Theta(|V|)$

$\Theta(|E|)$

```
DSF-Visita(u:vertice)
  colore[u] = Grigio
  d[u] = tempo = tempo + 1
  for each vertice v ∈ Adiac[u]
       do if colore[v] = Binaco
          then pred[v] = u
               DFS-Visit(v)
  colore[u] = Nero
  f[u] = tempo = tempo + 1
```

$\Theta(|E_u|)$

$\Theta(|E_{ric(u)}|)$

**Ma solo per vertici non ancora visitati**

# Tempo di esecuzione di DFS

```
DSF(G:grafo)
  for each vertice u ∈ V
        do  colore[u] = Binaco
            pred[u] = NIL
   tempo = 0
  for each vertice u ∈ V
        do if  colore[u] = Binaco
           then  DFS-Visita(u)
```

$\Theta(|V|+|E|)$