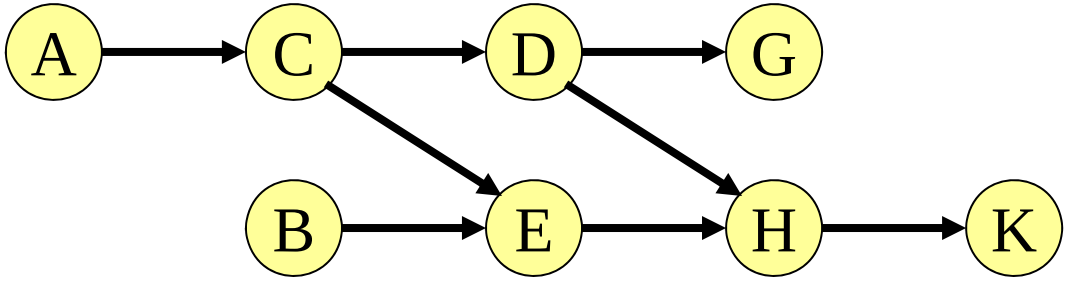
Topological sort: example

- Given a DAG of prerequisites for courses, a topological sort can be used to determine **an order** in which to take the courses
- (TS: DAG => sequence) (modified dfs)
- prints **reverse** topological order of a DAG from v

```
tsort(v)  {
   mark v visited
      for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```

Topological sort: example



A	C		
B	E		
C	D	E	
D	G	H	
E	H		
G			
H	K		
K			

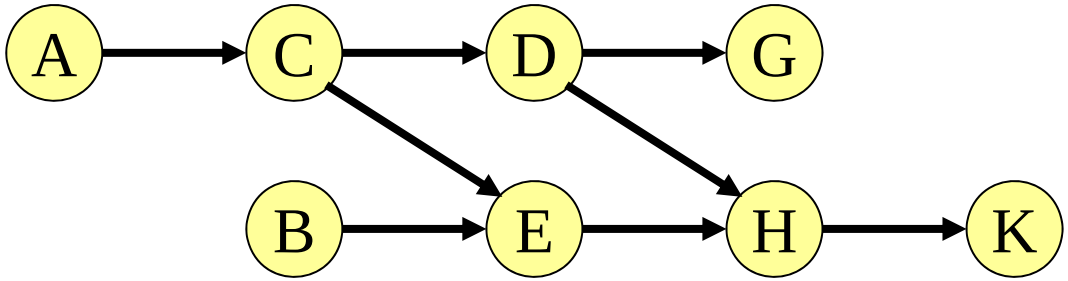
start: A

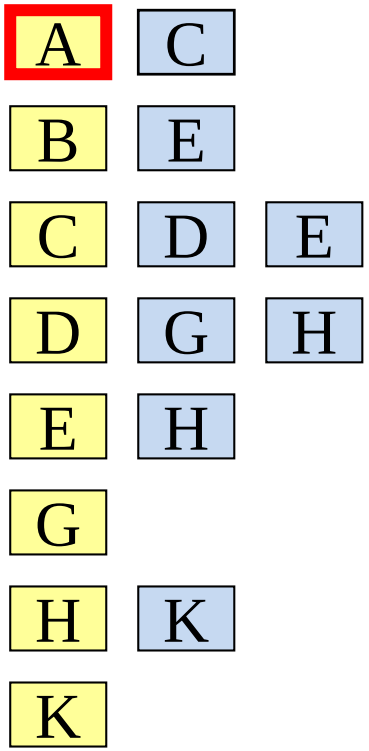
tsort(A) => G K H D E C A B

reverse => B A C E D H K G

Topological Sort example

```
tsort(v)    {
  A➔ mark v visited
      for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```





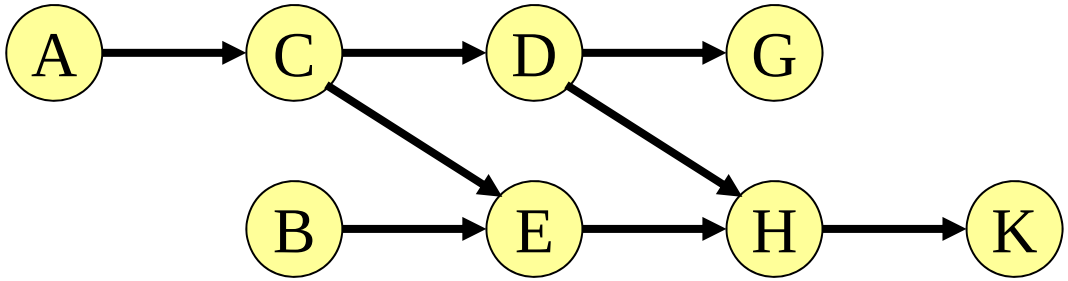
path: **A**

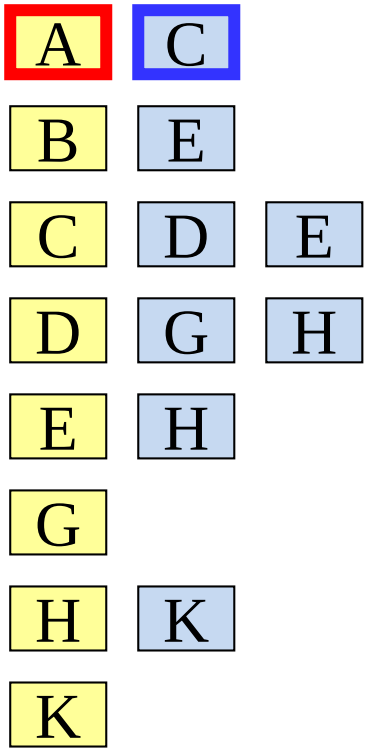
output:

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
 A➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





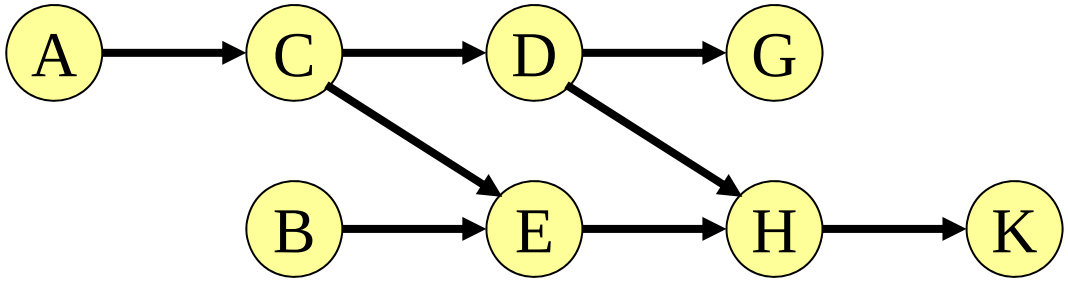
path: **A ➔ C**

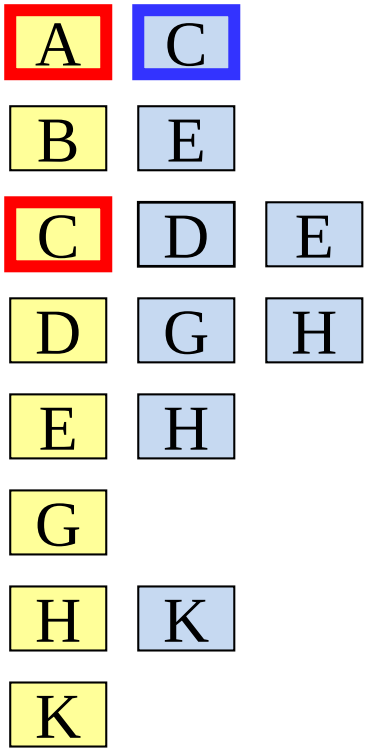
output:

reverse:

Topological Sort example

```
tsort(v)    {
  C ➔mark v visited
      for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```





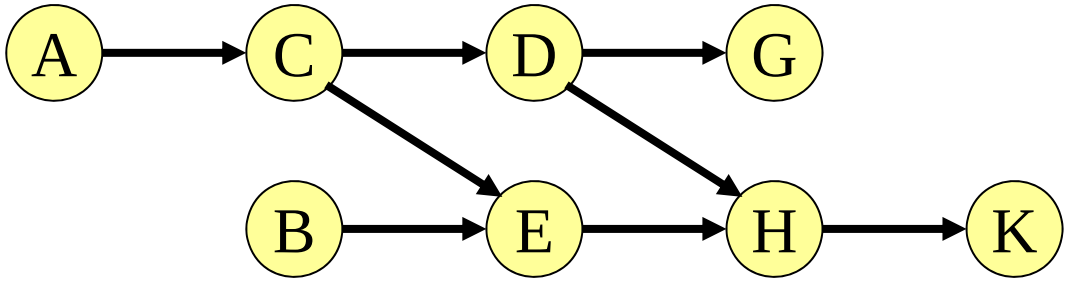
path: **A ➔ C**

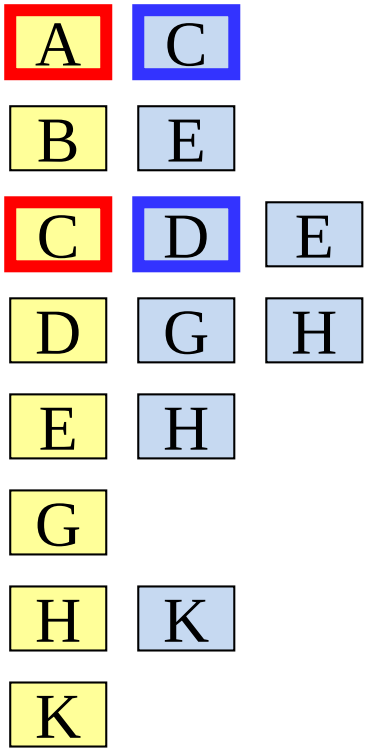
output:

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
 C ➔for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





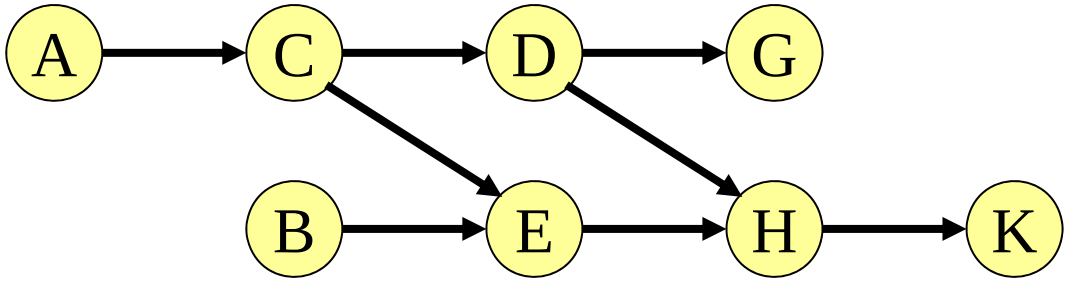
path: **A ➔ C ➔ D**

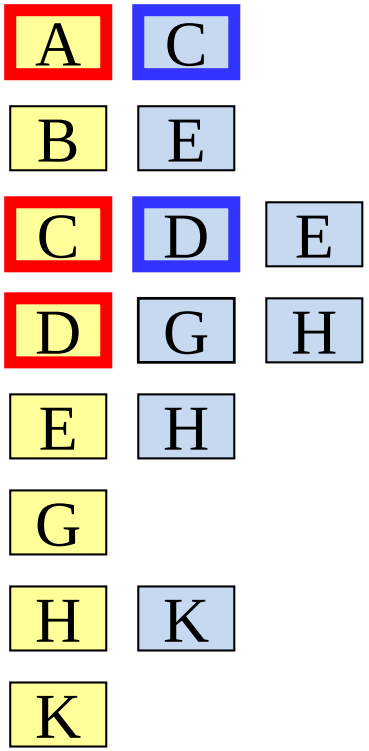
output:

reverse:

Topological Sort example

```
tsort(v)     {
  D ➔mark v visited
       for each w adjacent to v if w unvisited tsort(w)
       display(v)
       }
```





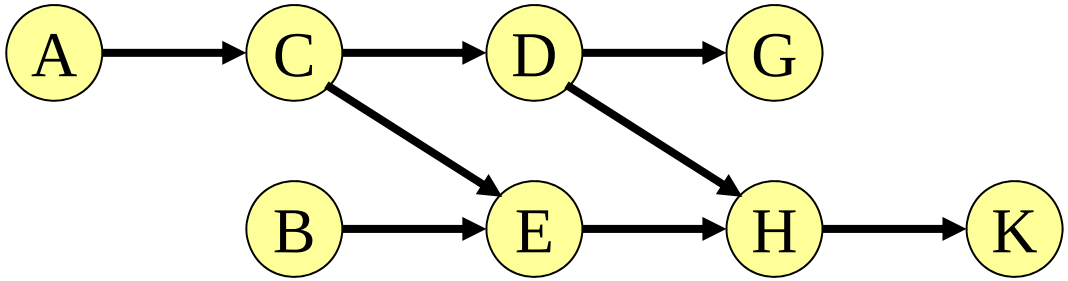
path: **A ➔ C ➔ D**

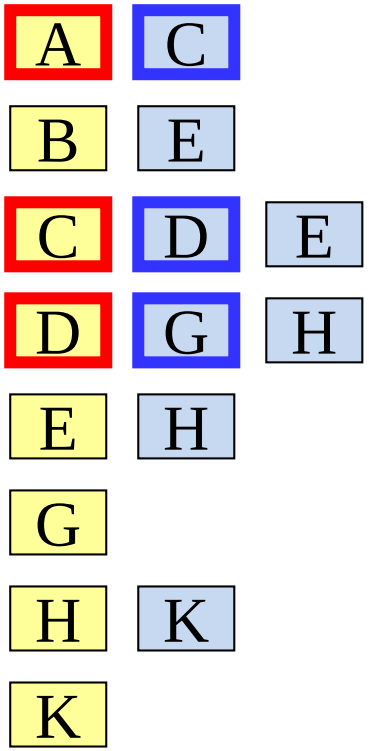
output:

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
 D ➔for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





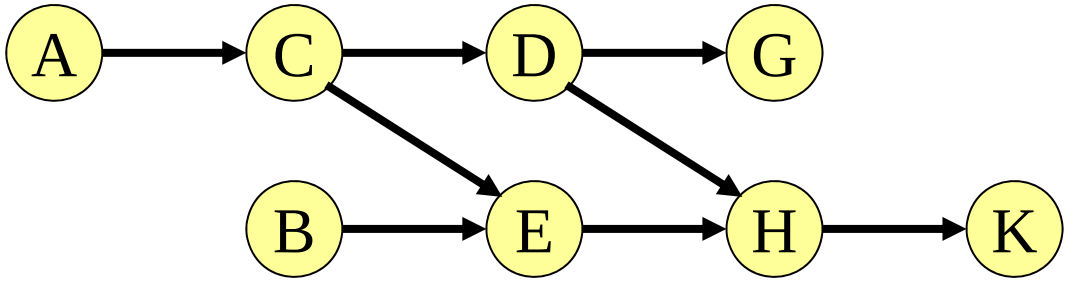
path: **A ➔ C ➔ D ➔ G**

output:

reverse:

Topological Sort example

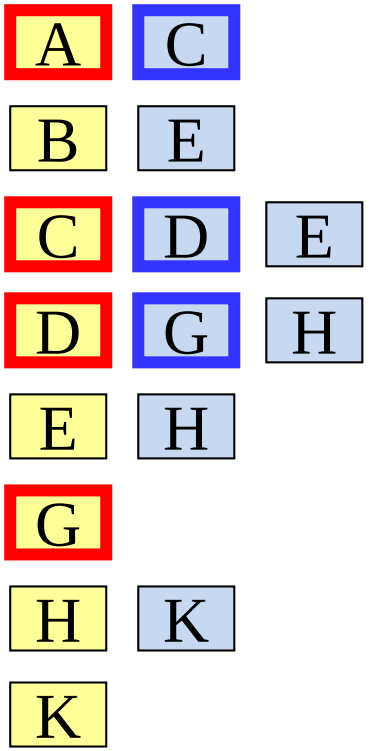
```
tsort(v)     {
 G ➔ mark v visited
       for each w adjacent to v if w unvisited tsort(w)
       display(v)
       }
```



path: **A ➔ C ➔ D ➔ G**

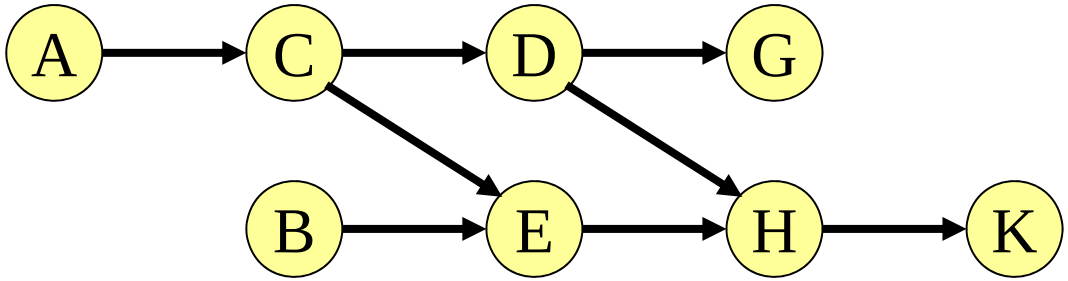
output:

reverse:



Topological Sort example

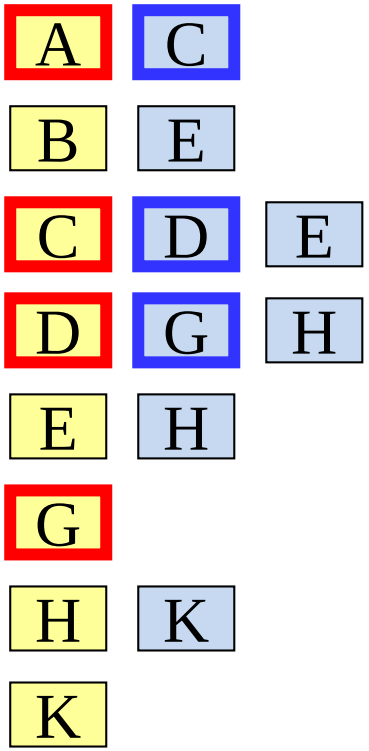
```
tsort(v)    {
        mark v visited
G ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```



path: **A ➔ C ➔ D ➔ G**

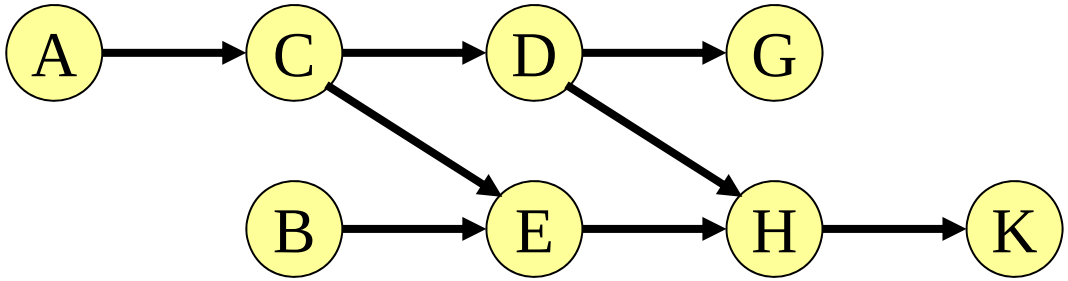
output:

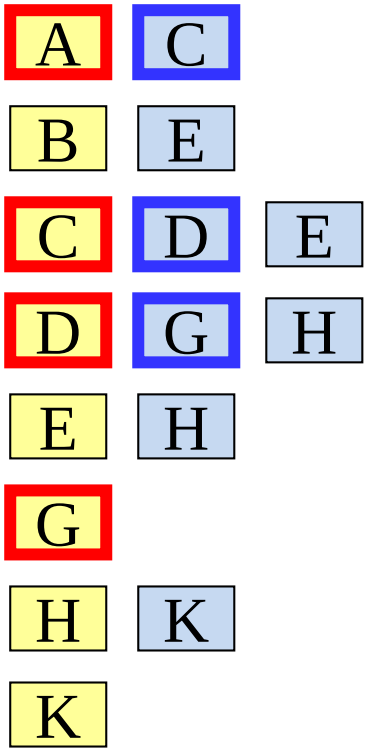
reverse:



Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 G ➔ display(v)
        }
```





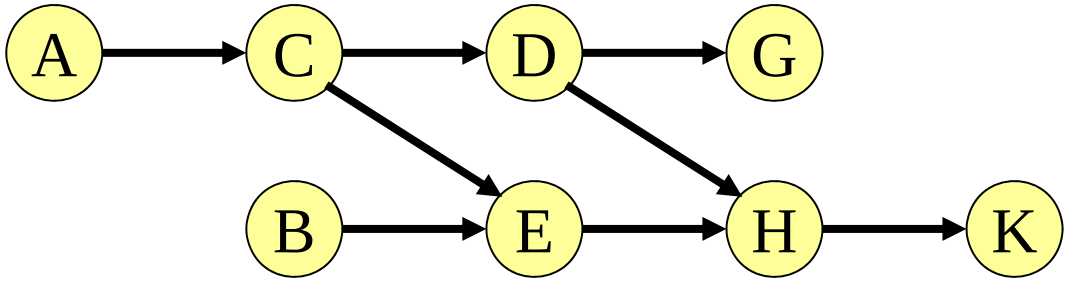
path: **A ➔ C ➔ D ➔ G**

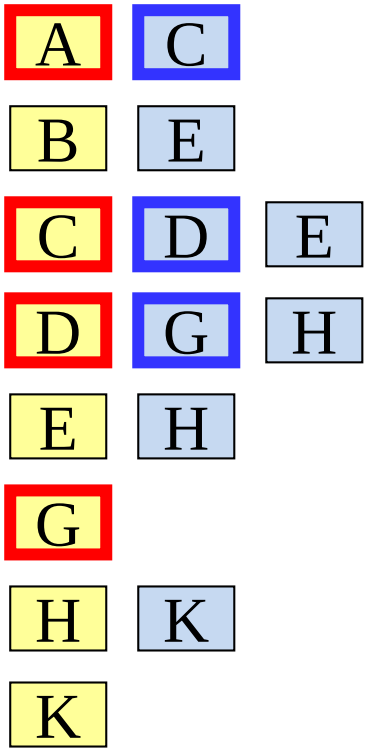
output: **G**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
D ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





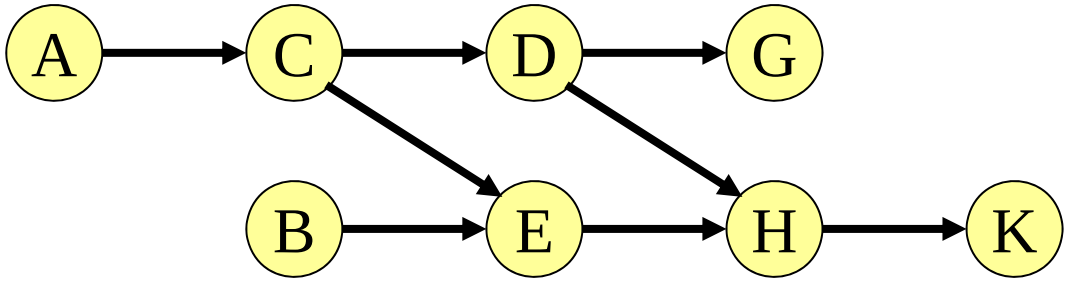
path: **A ➔ C ➔ D**

output: **G**

reverse:

Topological Sort example

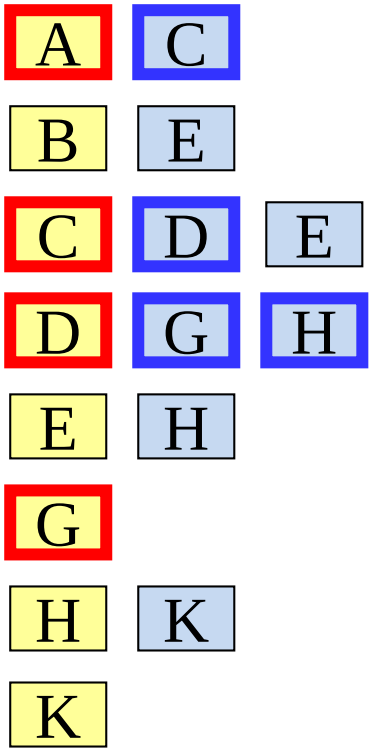
```
tsort(v)    {
        mark v visited
D ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```



path: **A ➔ C ➔ D ➔ H**

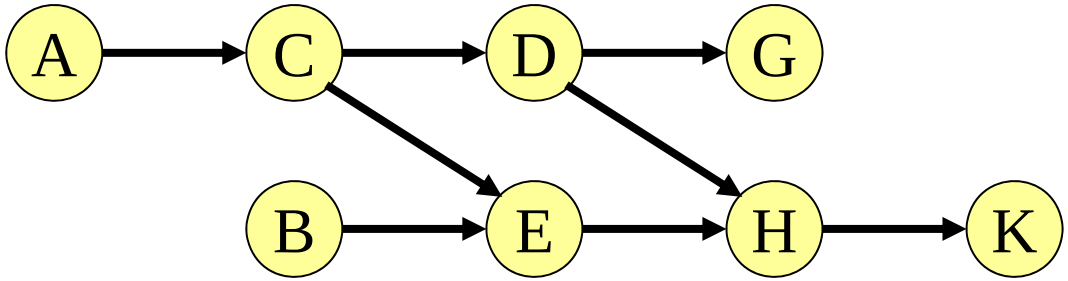
output: **G**

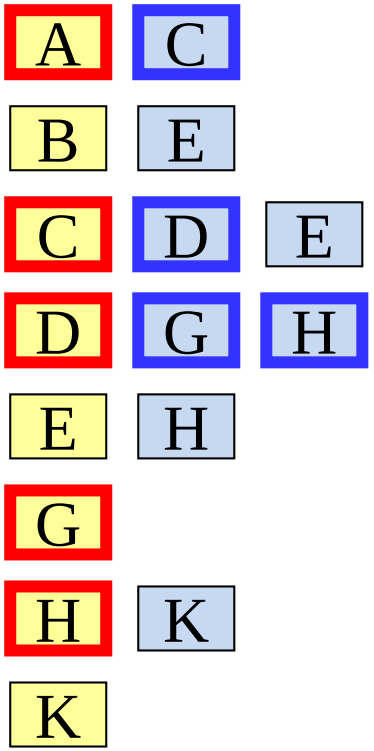
reverse:



Topological Sort example

```
tsort(v)    {
 H ➔ mark v visited
      for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```





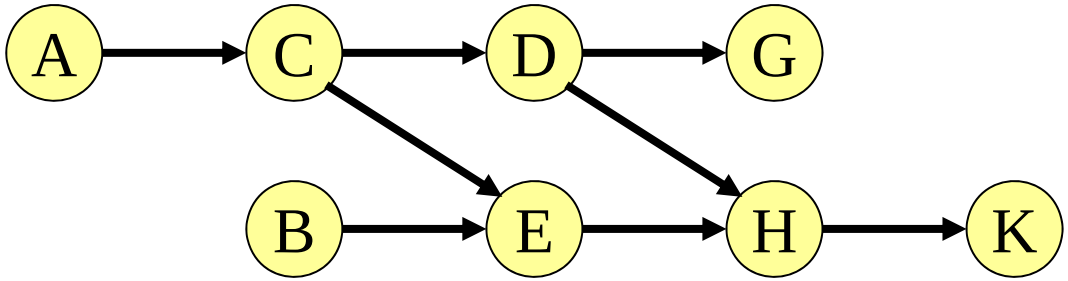
path: **A ➔ C ➔ D ➔ H**

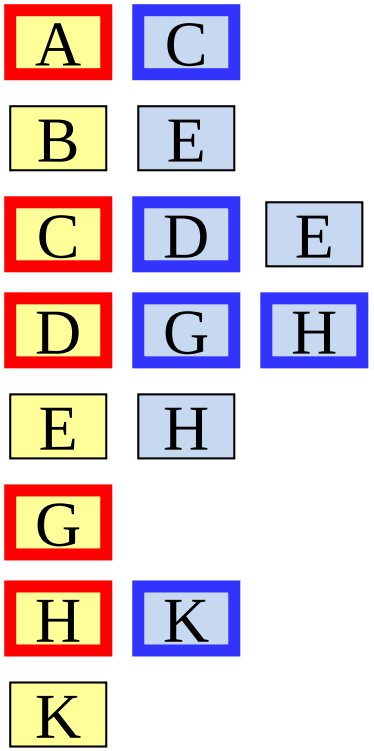
output: **G**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
H ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





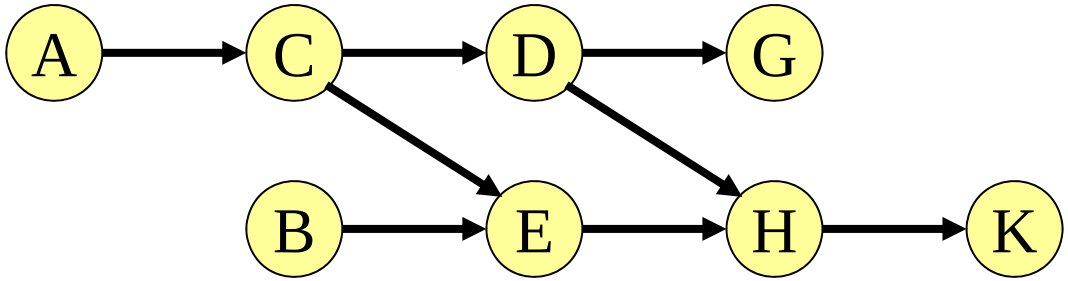
path: **A ➔ C ➔ D ➔ H ➔ K**

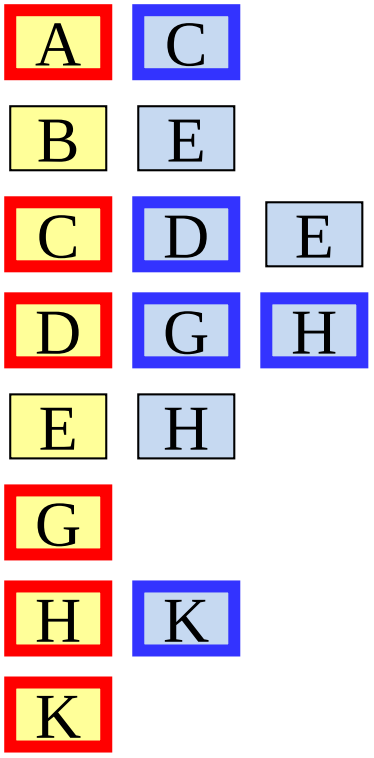
output: **G**

reverse:

Topological Sort example

```
tsort(v)     {
 K ➔ mark v visited
       for each w adjacent to v if w unvisited tsort(w)
       display(v)
       }
```





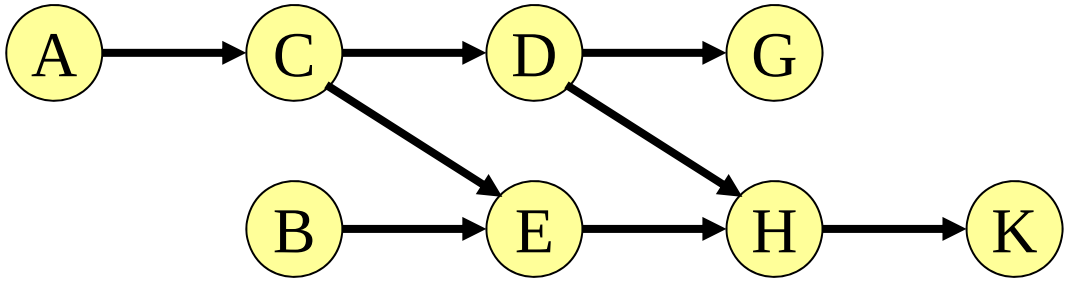
path: **A ➔ C ➔ D ➔ H ➔ K**

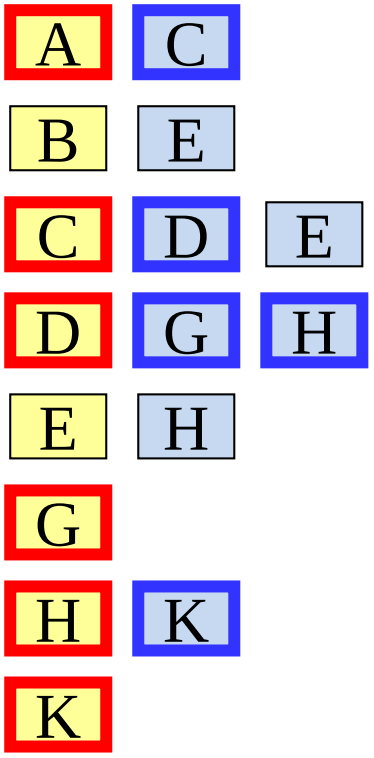
output: **G**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
K ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





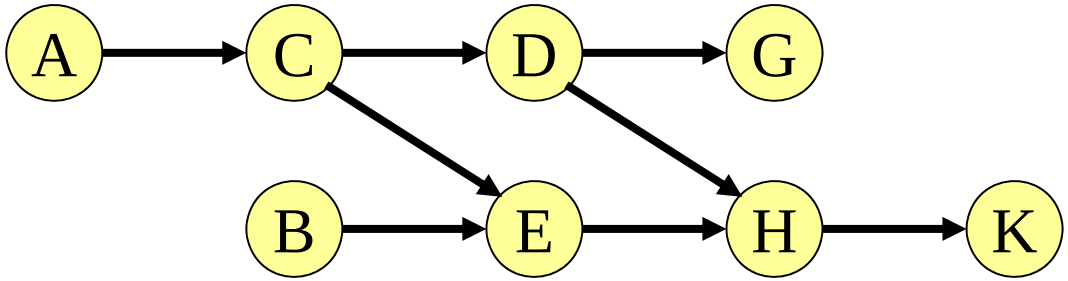
path: **A ➔ C ➔ D ➔ H ➔ K**

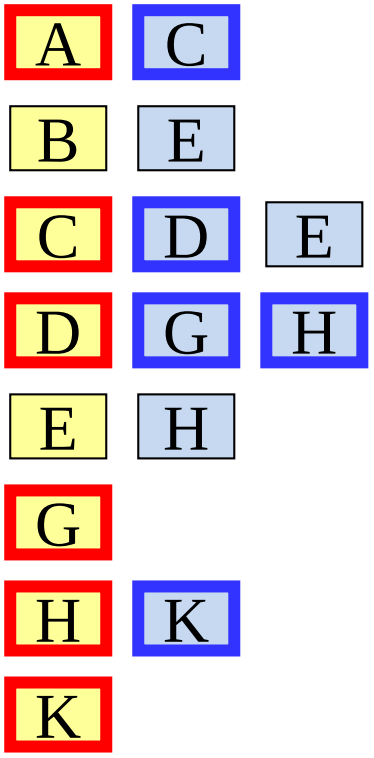
output: **G**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 K ➔ display(v)
        }
```





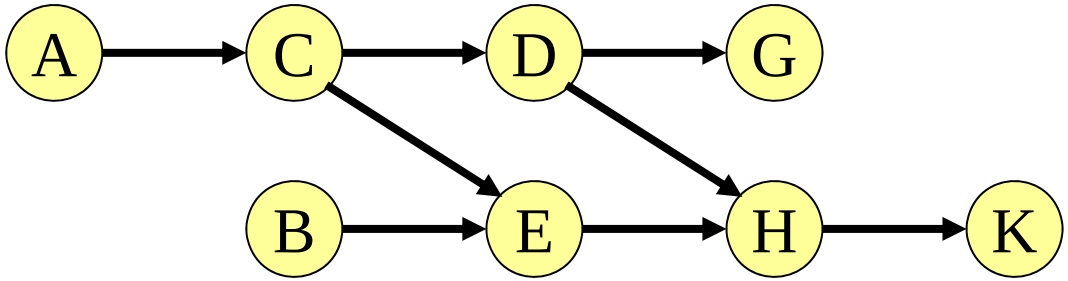
path: **A ➔ C ➔ D ➔ H ➔ K**

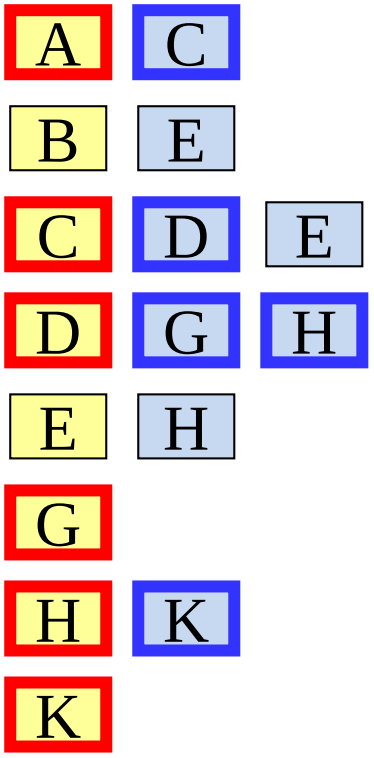
output: **G K**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
H ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





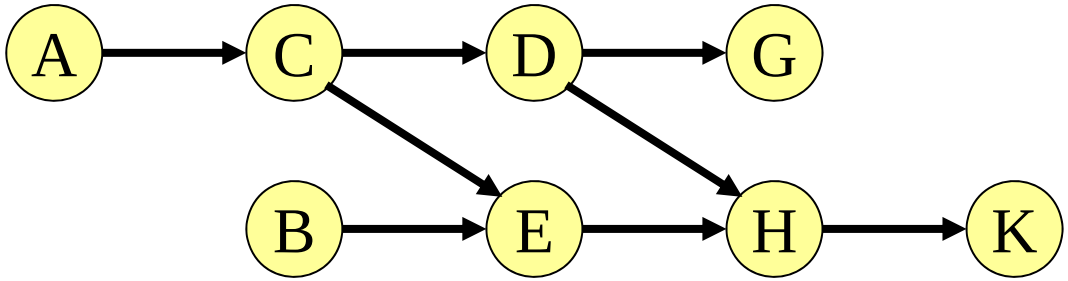
path: **A ➔ C ➔ D ➔ H**

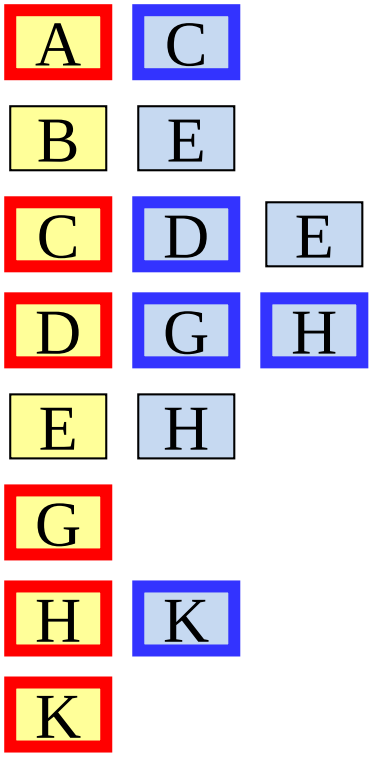
output: **G K**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 H ➔display(v)
        }
```





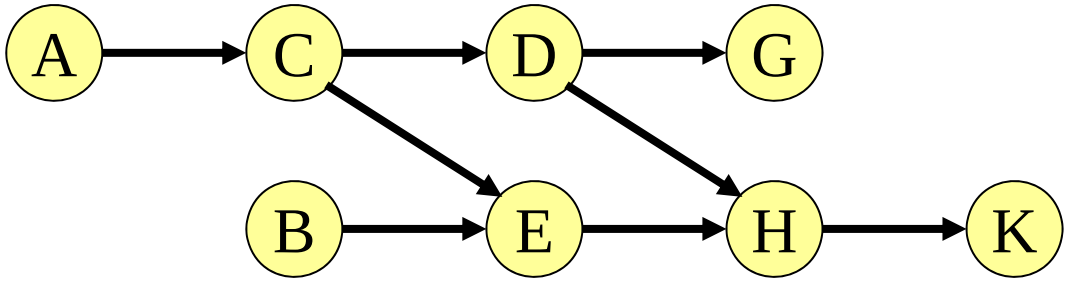
path: **A ➔ C ➔ D ➔ H**

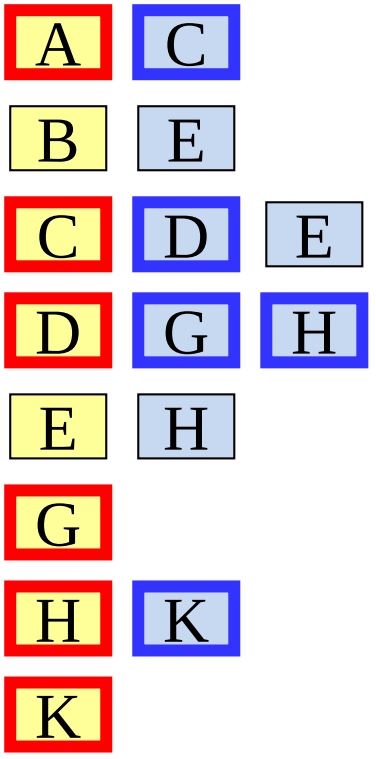
output: **G K H**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
D ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





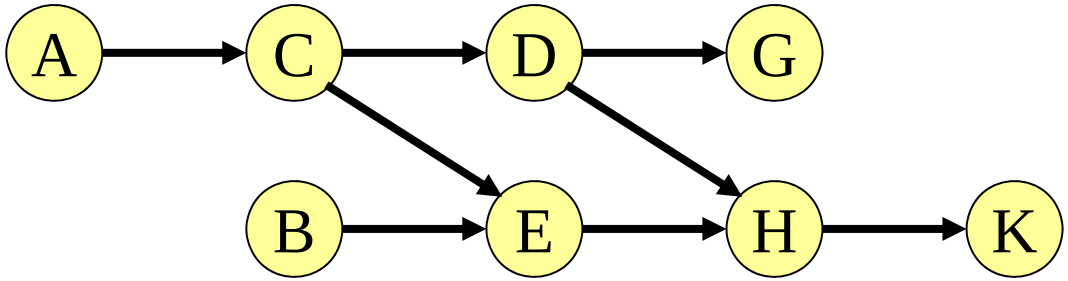
path: **A ➔ C ➔ D**

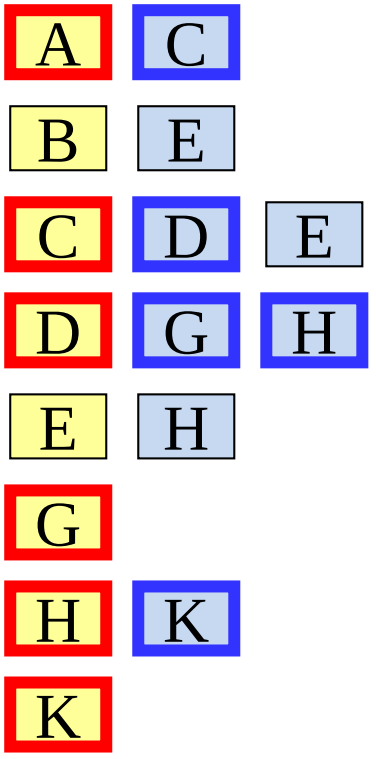
output: **G K H**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 D ➔ display(v)
        }
```





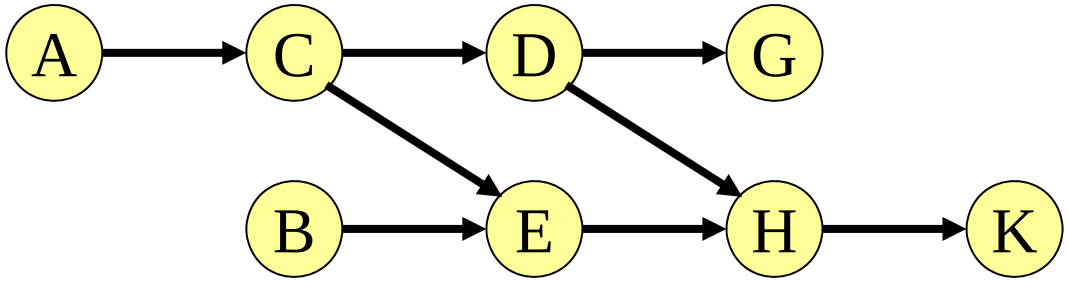
path: **A ➔ C ➔ D**

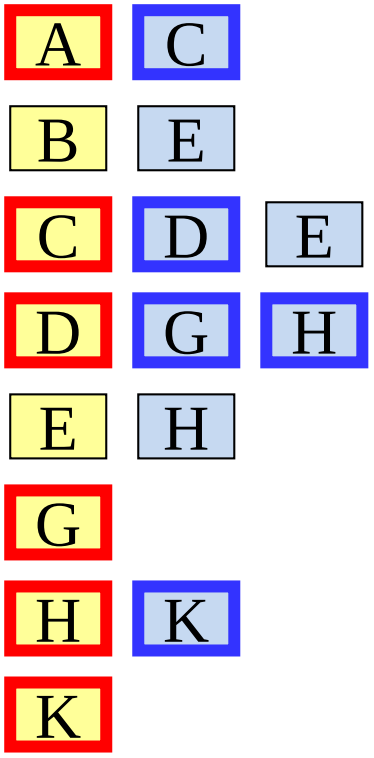
output: **G K H D**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
C➔      for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





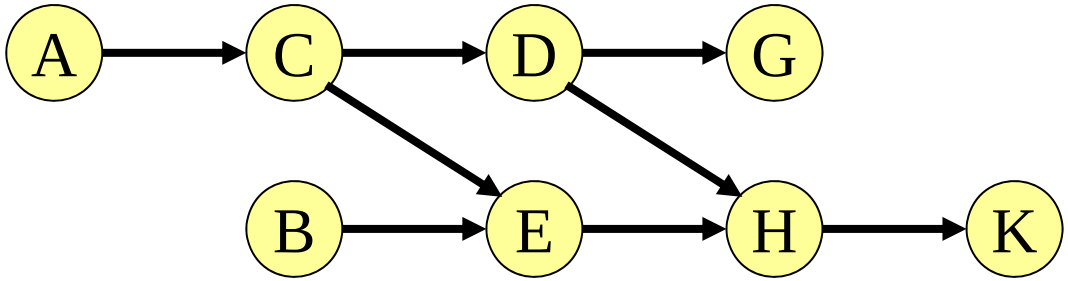
path: **A ➔ C**

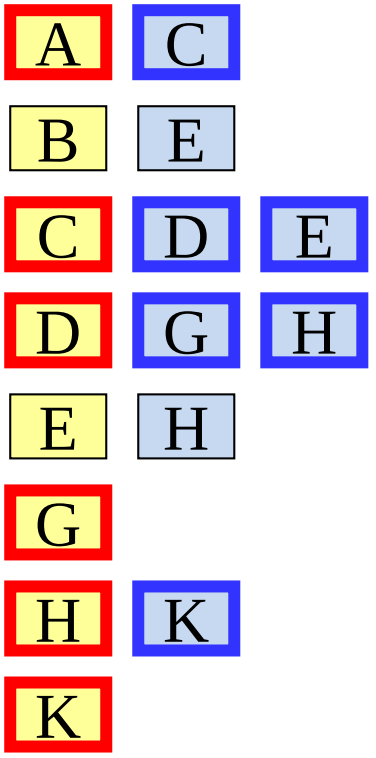
output: **G K H D**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
C➔      for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





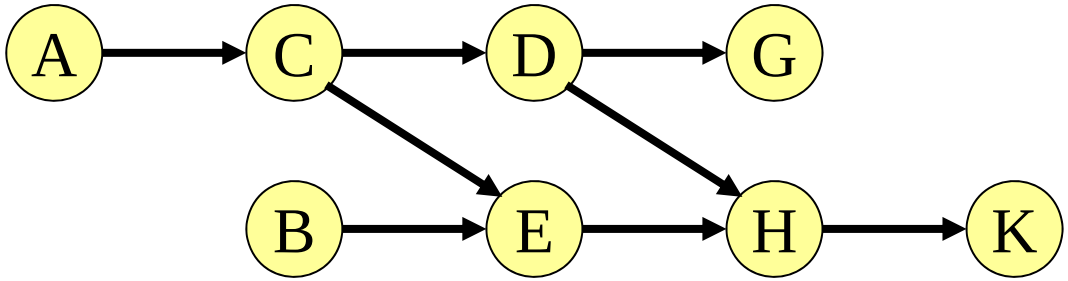
path: **A ➔ C ➔ E**

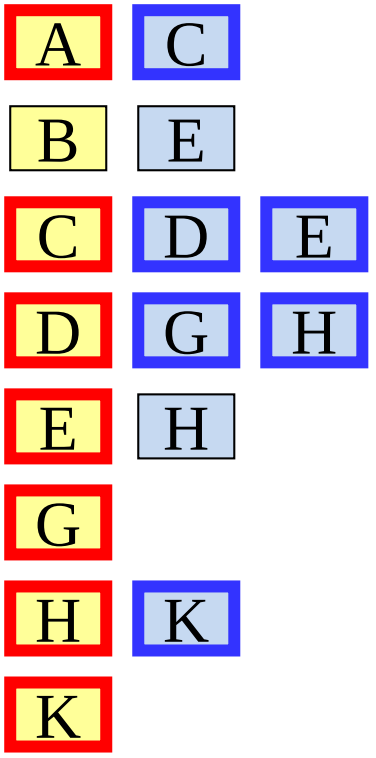
output: **G K H D**

reverse:

Topological Sort example

```
tsort(v)     {
 E ➔ mark v visited
       for each w adjacent to v if w unvisited tsort(w)
       display(v)
       }
```





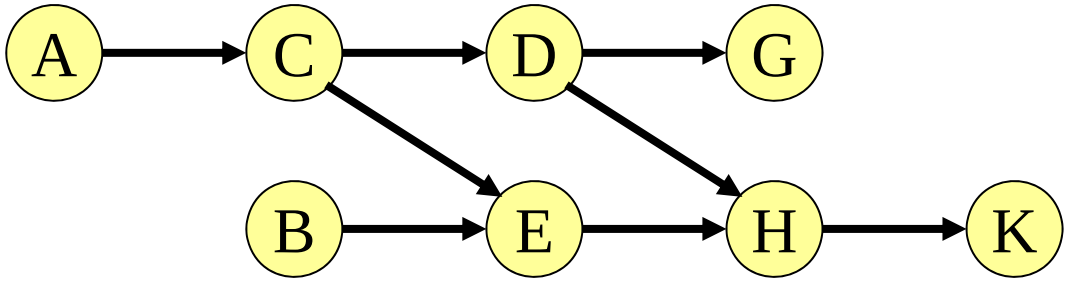
path: **A ➔ C ➔ E**

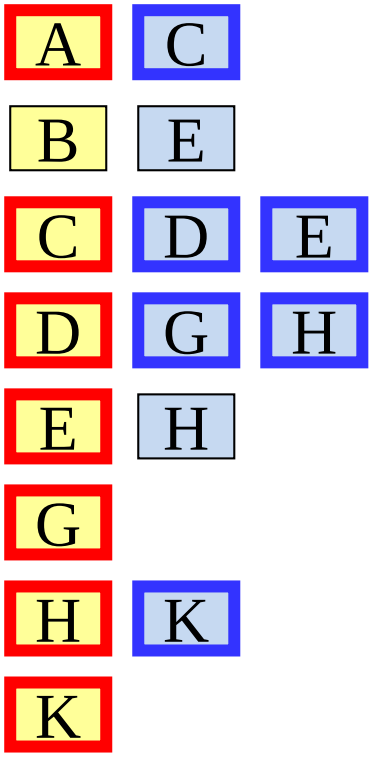
output: **G K H D**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
E ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





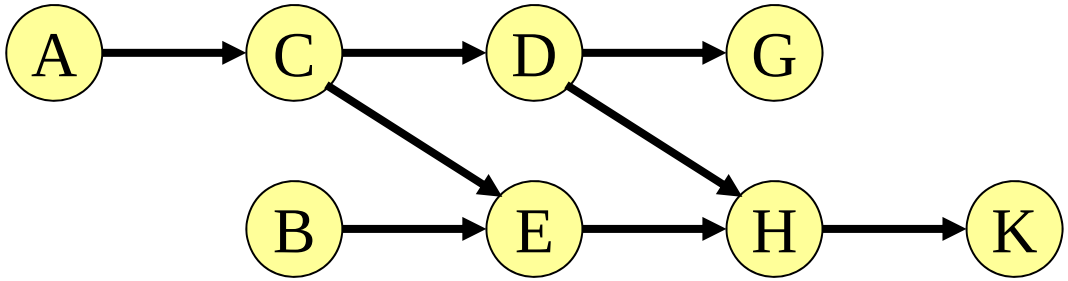
path: **A ➔ C ➔ E**

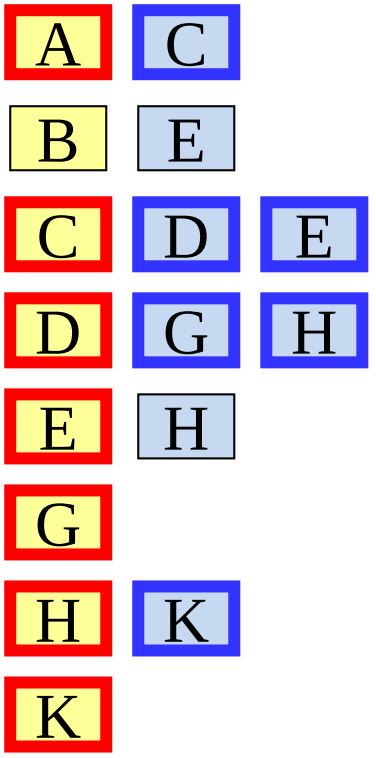
output: **G K H D**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 E ➔ display(v)
        }
```





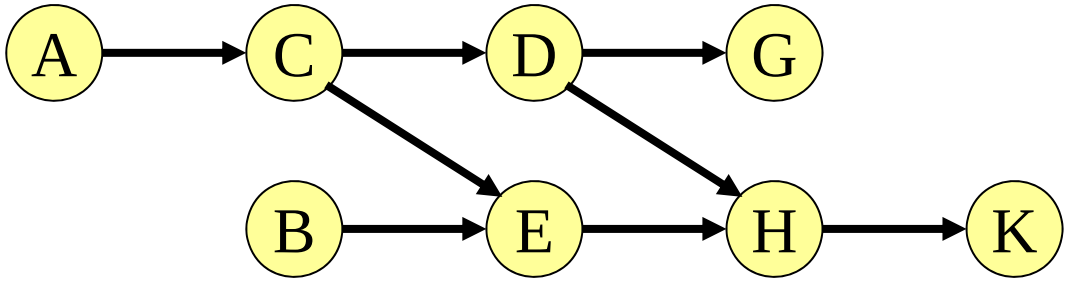
path: **A ➔ C ➔ E**

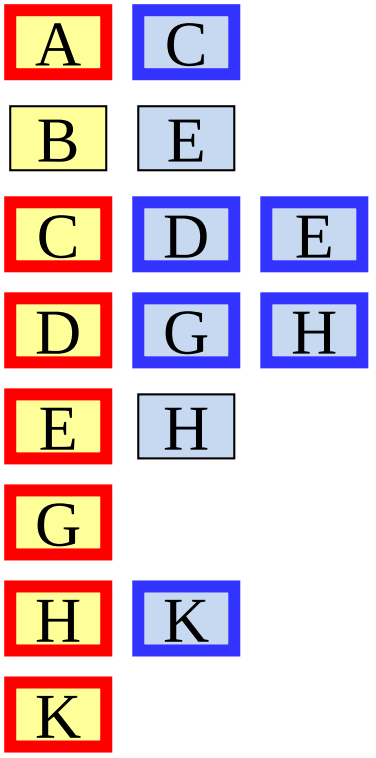
output: **G K H D E**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
C ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





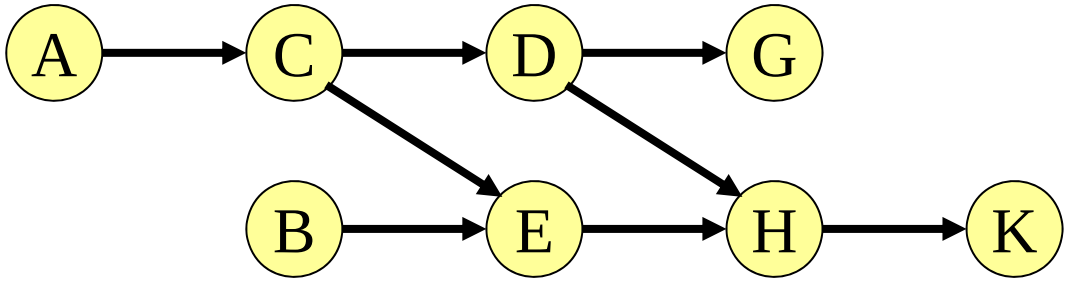
path: **A ➔ C**

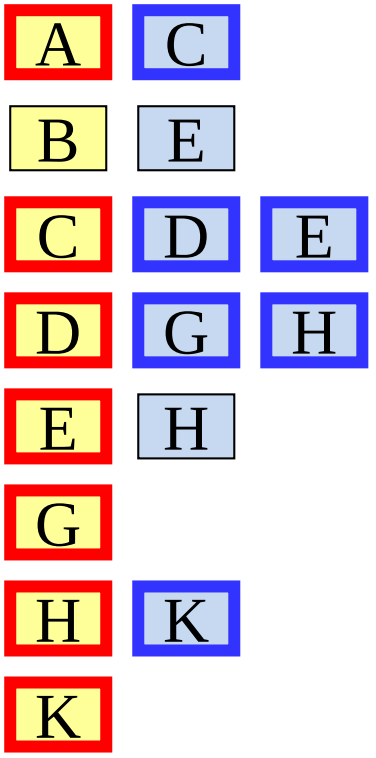
output: **G K H D E**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
 C ➔ display(v)
        }
```





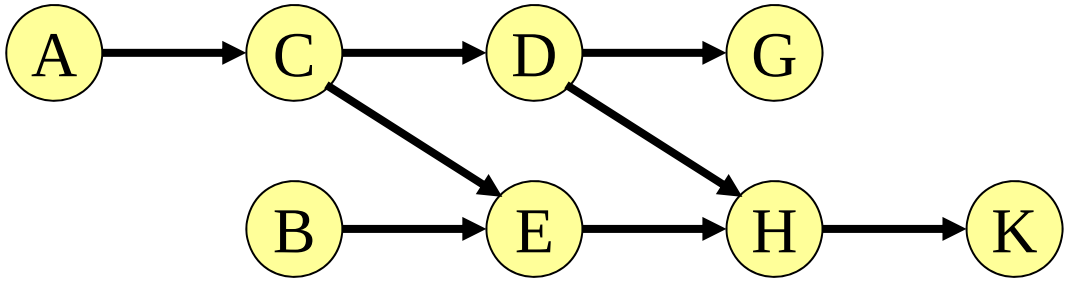
path: **A ➔ C**

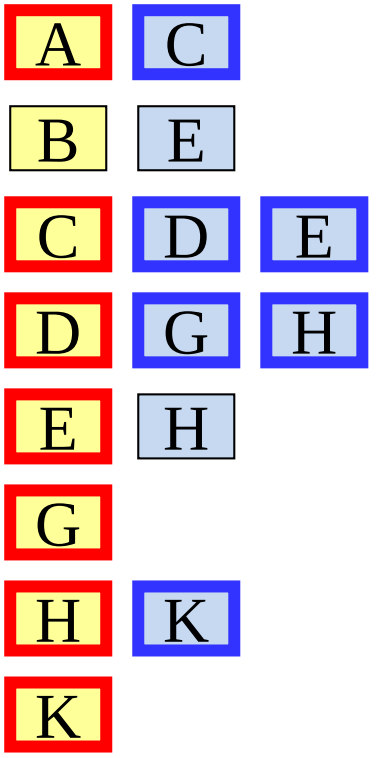
output: **G K H D E C**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
A ➔ for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





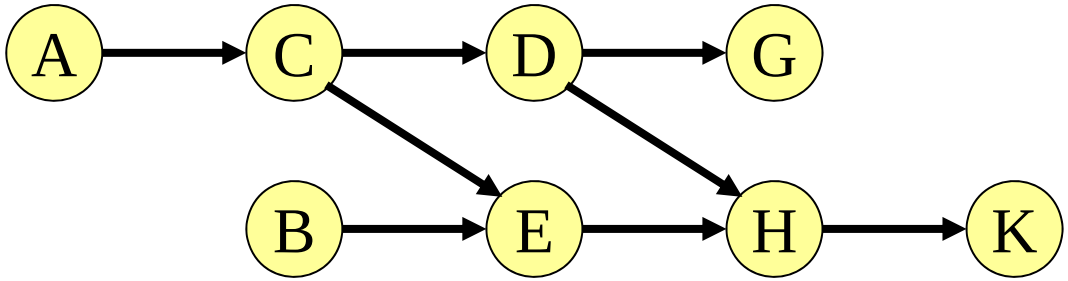
path: **A**

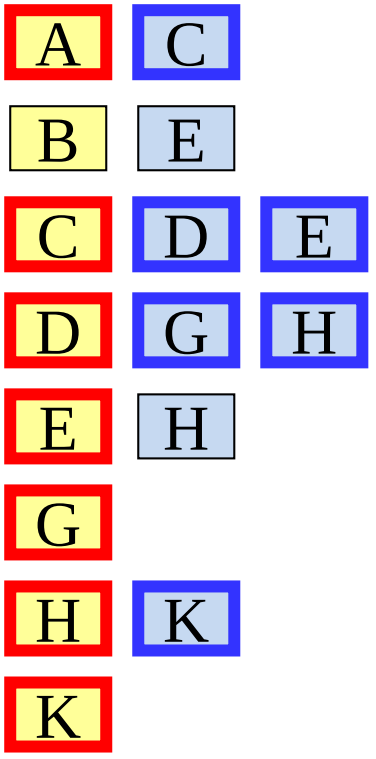
output: **G K H D E C**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
A ➔ display(v)
        }
```





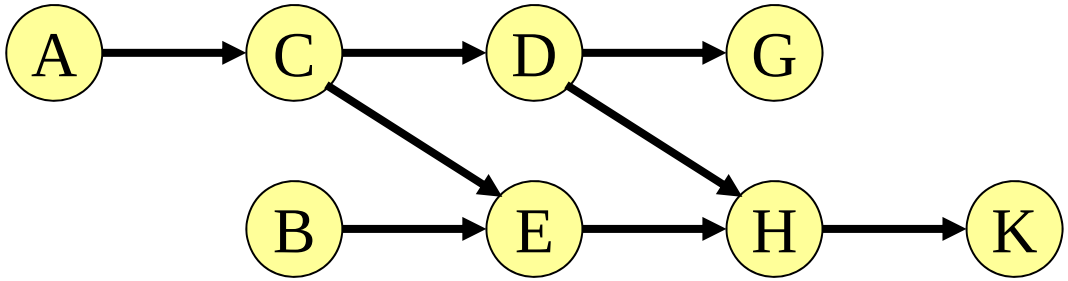
path: **A**

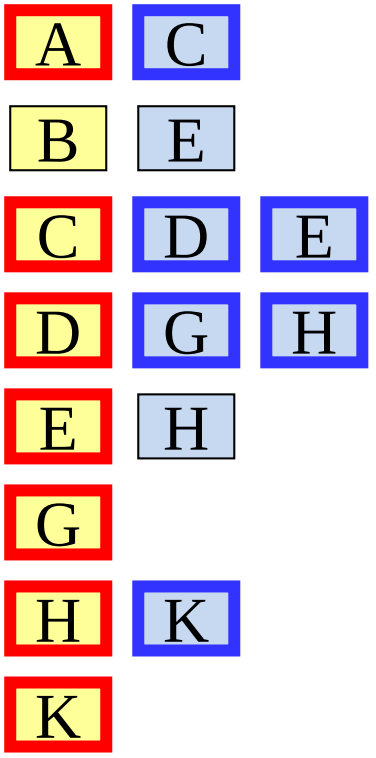
output: **G K H D E C A**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





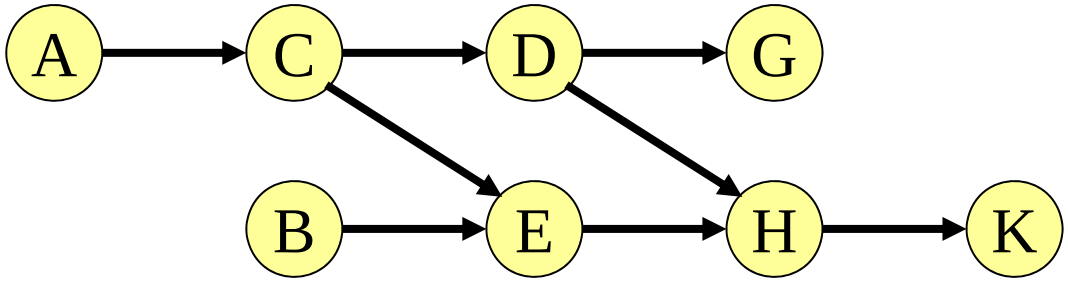
path:

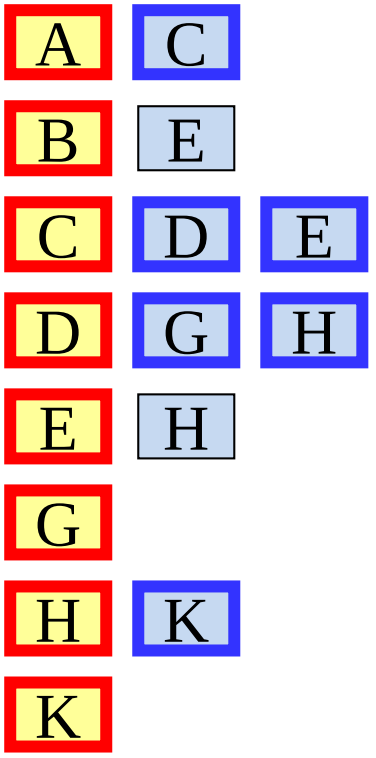
output: **G K H D E C A**

reverse:

Topological Sort example

```
tsort(v)    {
 B ➔mark v visited
      for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```





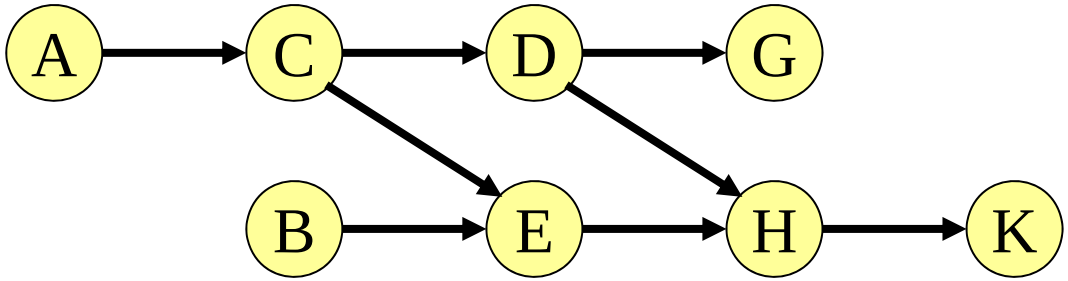
path: **B**

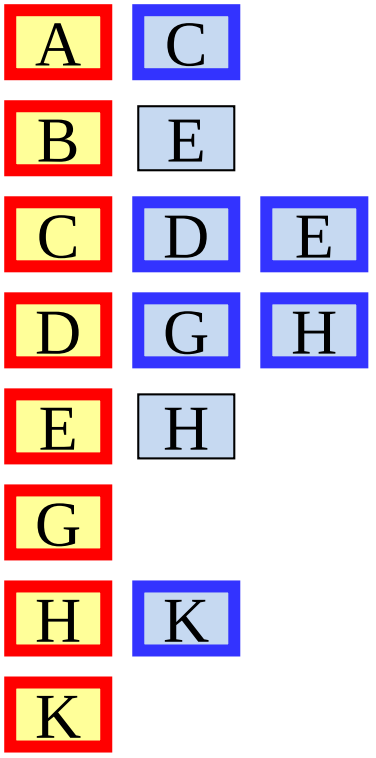
output: **G K H D E C A**

reverse:

Topological Sort example

```
tsort(v)    {
      mark v visited
 B ➔for each w adjacent to v if w unvisited tsort(w)
      display(v)
      }
```





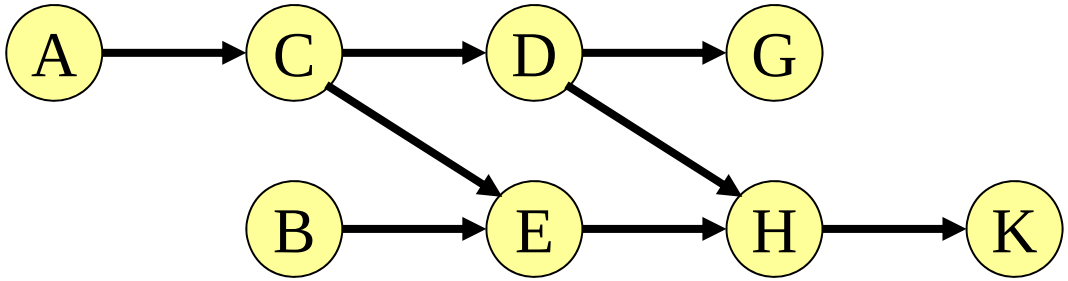
path: **B**

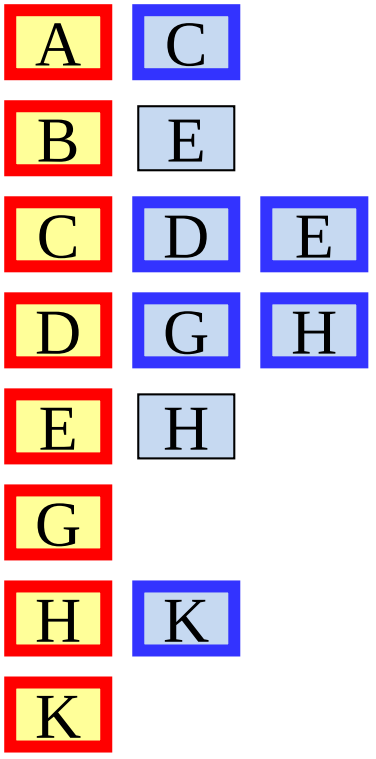
output: **G K H D E C A**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
B ➔ display(v)
        }
```





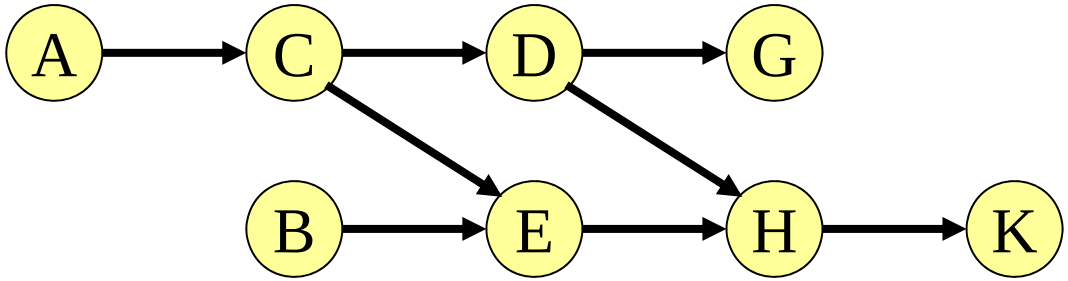
path: **B**

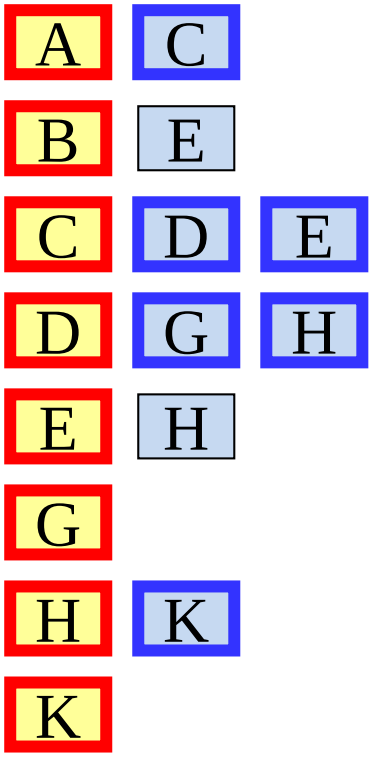
output: **G K H D E C A B**

reverse:

Topological Sort example

```
tsort(v)    {
        mark v visited
        for each w adjacent to v if w unvisited tsort(w)
        display(v)
        }
```





path:

output: **G K H D E C A B**

reverse: **B A C E D H K G**