Plain algo and code

```
title: "Floyd-Warshall" (algo only)
parameters: ("V", "E", "w") (algo only)
```

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
 3
            \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                                   // edge weights
 4 For v in V:
            \operatorname{dist}[v,v] \leftarrow 0
                                                                                   // base case
 5
 6
 7 For k \leftarrow 1 to |V|:
 8
            For i \leftarrow 1 to |V|:
                  For j \leftarrow 1 to |V|:
 9
                        // if new path is shorter, reduce distance
10
                        If dist[i, j] > dist[i, k] + dist[k, j]:
11
                               \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
12
13
14 Return dist
```

```
1 def floyd_warshall(G):
     # let G be an adjacency matrix
 2
 3
     dist = G
 4
     for k in range(len(G)):
 5
       for i in range(len(G)):
6
         for j in range(len(G)):
7
            if dist[i][j] > dist[i][k] + dist[k][j]:
 8
              dist[i][j] = dist[i][k] + dist[k][j]
10
     return dist
11
12
```

Basic styling parameters

```
fill: none
stroke: 2pt + black
radius: 10pt
row-gutter: 8pt
column-gutter: 8pt
inset: 15pt
indent-size: 12pt (algo only)
indent-guides: 1pt + gray
indent-guides-offset: 4pt
comment-prefix: [#sym.triangle] (algo only)
```

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
         \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                            △ edge weights
 4 For v in V:
         \operatorname{dist}[v,v] \leftarrow 0
                                                                            △ base case
 6
 7 For k \leftarrow 1 to |V|:
         For i \leftarrow 1 to |V|:
 9
             For j \leftarrow 1 to |V|:
                \triangle if new path is shorter, reduce distance
10
                If dist[i, j] > dist[i, k] + dist[k, j]:
11
                    \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
12
13
14 Return dist
```

```
1 def floyd_warshall(G):
 2
     # let G be an adjacency matrix
 3
     dist = G
 4
 5
     for k in range(len(G)):
       for i in range(len(G)):
 6
 7
         for j in range(len(G)):
          if dist[i][j] > dist[i][k] + dist[k][j]:
 8
             dist[i][j] = dist[i][k] + dist[k][j]
10
     return dist
11
12
```

Empty bodies

code with empty raw text

code with empty raw block

${\bf code} \ with \ non-sequence \ raw \ block$

```
1 def floyd_warshall(G):
   # let G be an adjacency matrix
2
    dist = G
3
4
5
   for k in range(len(G)):
      for i in range(len(G)):
6
7
        for j in range(len(G)):
8
           if dist[i][j] > dist[i][k] + dist[k][j]:
             dist[i][j] = dist[i][k] + dist[k][j]
10
11
     return dist
12
```

Indent guides with line wrapping

indent-guides: 1pt + black

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
          \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                                               // edge weights
 3
    For v in V:
 4
          \operatorname{dist}[v,v] \leftarrow 0
 5
                                                                                               // base case
 6
     For k \leftarrow 1 to |V|:
 7
          For i \leftarrow 1 to |V|:
 8
 9
               For j \leftarrow 1 to |V|:
                    // if new path is shorter, reduce distance
10
                     If dist[i, j] > dist[i, k] + dist[k, j]:
11
                          \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
                          13
                          blah blah blah blah
14
15 Return dist
```

```
1 def floyd_warshall(G):
2
    # let G be an adjacency matrix
    dist = G
3
4
5
    for k in range(len(G)):
      for i in range(len(G)):
6
        for j in range(len(G)):
7
         if dist[i][j] > dist[i][k] + dist[k][j]:
8
           dist[i][j] = dist[i][k] + dist[k][j]
9
           10
   blah blah blah
12
13
    return dist
```

code indent guides with custom tab size

indent-guides: 1pt + black

tab-size: 2

```
1 def floyd_warshall(
       G
2
3
     # let G be an adjacency matrix
4
     dist = G
5
6
     for k in range(len(G)):
7
      for i in range(len(G)):
8
         for j in range(len(G)):
9
          if dist[i][j] > dist[i][k] + dist[k][j]:
10
             dist[i][j] = dist[i][k] + dist[k][j]
12
13
      return dist
14
```

No line numbers

line-numbers: false

```
 \begin{array}{c} {\bf FLOYD\text{-}Warshall}(V,E,w)\colon \\ {\bf Let}\ {\rm dist}[u,v] \leftarrow \infty\ {\bf for}\ u,v\ {\bf in}\ V \\ {\bf For}\ (u,v)\ {\bf in}\ E\colon \\ & {\rm dist}[u,v] \leftarrow w(u,v) & //\ {\rm edge}\ {\rm weights} \\ {\bf For}\ v\ {\bf in}\ V\colon \\ & {\rm dist}[v,v] \leftarrow 0 & //\ {\rm base}\ {\rm case} \\ \\ {\bf For}\ k \leftarrow 1\ {\bf to}\ |V|\colon \\ & {\bf For}\ i \leftarrow 1\ {\bf to}\ |V|\colon \\ & {\bf For}\ j \leftarrow 1\ {\bf to}\ |V|\colon \\ & //\ {\rm if}\ {\rm new}\ {\rm path}\ {\rm is}\ {\rm shorter},\ {\rm reduce}\ {\rm distance} \\ & {\bf If}\ {\rm dist}[i,j] > {\rm dist}[i,k] + {\rm dist}[k,j]\colon \\ & {\rm dist}[i,j] \leftarrow {\rm dist}[i,k] + {\rm dist}[k,j] \\ \\ {\bf Return}\ {\rm dist} \end{array}
```

```
def floyd_warshall(G):
    # let G be an adjacency matrix
    dist = G

for k in range(len(G)):
    for i in range(len(G)):
        for j in range(len(G)):
        if dist[i][j] > dist[i][k] + dist[k][j]:
            dist[i][j] = dist[i][k] + dist[k][j]
return dist
```

algo without keywords

strong-keywords: false

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
            \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                                // edge weights
 3
     For v in V:
 4
            \mathrm{dist}[v,v] \leftarrow 0
                                                                                // base case
 5
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo with custom keywords

keywords: ("in", "to", "hello world")

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
           \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                              // edge weights
 3
     For v in V:
 4
           \mathrm{dist}[v,v] \leftarrow 0
 5
                                                                              // base case
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                            \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14
     blah blah hello world blah blah
15
     Return dist
```

algo without title

title: none

```
(V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
            \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                                // edge weights
 3
 4 For v in V:
            \mathrm{dist}[v,v] \leftarrow 0
                                                                                // base case
 5
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo without parameters

parameters: ()

```
FLOYD-WARSHALL():
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
           \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                               // edge weights
 3
 4 For v in V:
           \mathrm{dist}[v,v] \leftarrow 0
                                                                               // base case
 5
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo without header

title: none parameters: ()

```
1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
            \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                                 // edge weights
 3
 4 For v in V:
           \mathrm{dist}[v,v] \leftarrow 0
 5
                                                                                 // base case
 6
     For k \leftarrow 1 to |V|:
 7
           For i \leftarrow 1 to |V|:
 8
 9
                  For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                              \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo with content-type parameters

parameters: ([#text(blue, [V])], [#text(red, [E])], [#text(green, [w])])

```
FLOYD-WARSHALL(V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
           \mathrm{dist}[u,v] \leftarrow w(u,v)
 3
                                                                               // edge weights
    For v in V:
 4
           \mathrm{dist}[v,v] \leftarrow 0
 5
                                                                               // base case
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo with content-type title

title: [#set text(red);Floyd-Warshall]

```
Floyd-Warshall():
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
            \mathrm{dist}[u,v] \leftarrow w(u,v)
                                                                                // edge weights
 3
    For v in V:
 4
            \mathrm{dist}[v,v] \leftarrow 0
                                                                                // base case
 5
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
 9
                 For j \leftarrow 1 to |V|:
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

algo with custom header

```
Floyd-Warshall Algorithm
       Inputs: graph G = (V, E)
                    weight function w:E \to \mathbb{R}
     Outputs: distance matrix dist
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
           \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                             // edge weights
 3
    For v in V:
 4
           \operatorname{dist}[v,v] \leftarrow 0
                                                                             // base case
 5
 6
     For k \leftarrow 1 to |V|:
 7
           For i \leftarrow 1 to |V|:
 8
                For j \leftarrow 1 to |V|:
 9
                      // if new path is shorter, reduce distance
10
11
                      If dist[i, j] > dist[i, k] + dist[k, j]:
                            \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

Text styling

```
main-text-styles: (fill: green)
line-number-styles: (fill: red)
comment-styles: (fill: blue) (algo only)
```

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
    For (u, v) in E:
            \operatorname{dist}[u,v] \leftarrow w(u,v)
 3
                                                                                   // edge weights
 4 For v in V:
            \operatorname{dist}[v,v] \leftarrow 0
 5
                                                                                   // base case
 6
 7
      For k \leftarrow 1 to |V|:
            For i \leftarrow 1 to |V|:
 8
 9
                  For j \leftarrow 1 to |V|:
                        // if new path is shorter, reduce distance
10
                        If dist[i, j] > dist[i, k] + dist[k, j]:
11
                              \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
12
13
14 Return dist
```

```
def floyd_warshall(G):
 2
     # let G be an adjacency matrix
     dist = G
 3
 4
 5
     for k in range(len(G)):
6
      for i in range(len(G)):
          for j in range(len(G)):
7
8
            if dist[i][j] > dist[i][k] + dist[k][j]:
              dist[i][j] = dist[i][k] + dist[k][j]
10
      return dist
11
12
```

Indent guides with big main text

indent-guides: 1pt + black main-text-styles: (size: 15pt)

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 <sup>2</sup> For (u, v) in E:
          \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                                     // edge weights
 4 For v in V:
          \operatorname{dist}[v,v] \leftarrow 0
                                                                                     // base case
 6
    For k \leftarrow 1 to |V|:
 7
          For i \leftarrow 1 to |V|:
               For j \leftarrow 1 to |V|:
 9
                   // if new path is shorter, reduce distance
10
                   If dist[i, j] > dist[i, k] + dist[k, j]:
11
                         \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
12
13
14 Return dist
```

```
1 def floyd warshall(G):
    # let G be an adjacency matrix
2
    dist = G
3
4
    for k in range(len(G)):
5
      for i in range(len(G)):
6
        for j in range(len(G)):
7
           if dist[i][j] > dist[i][k] + dist[k]
  [j]:
             dist[i][j] = dist[i][k] + dist[k]
12
13
    return dist
```

Indent guides with big line numbers

indent-guides: 1pt + black line-number-styles: (size: 15pt)

```
FLOYD-WARSHALL(V, E, w):
  1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
  2 For (u, v) in E:
  3
             \operatorname{dist}[u,v] \leftarrow w(u,v)
                                                                                   // edge weights
  4 For v in V:
  5
             \operatorname{dist}[v,v] \leftarrow 0
                                                                                   // base case
  6
  7
      For k \leftarrow 1 to |V|:
  8
             For i \leftarrow 1 to |V|:
  9
                   For j \leftarrow 1 to |V|:
10
                         // if new path is shorter, reduce distance
                         If dist[i, j] > dist[i, k] + dist[k, j]:
11
12
                               \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
13
14 Return dist
```

```
def floyd_warshall(G):
 2
      # let G be an adjacency matrix
 3
      dist = G
 4
 5
      for k in range(len(G)):
 6
        for i in range(len(G)):
          for j in range(len(G)):
 7
 8
           if dist[i][j] > dist[i][k] + dist[k][j]:
              dist[i][j] = dist[i][k] + dist[k][j]
10
11
      return dist
12
```

algo indent guides with big comments

indent-guides: 1pt + black comment-styles: (size: 15pt)

```
FLOYD-WARSHALL (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
      For (u, v) in E:
                                                                                                            // edge weights
 3
             \operatorname{dist}[u,v] \leftarrow w(u,v)
      For v in V:
 4
                                                                                                            // base case
 5
             \operatorname{dist}[v,v] \leftarrow 0
 6
      For k \leftarrow 1 to |V|:
 7
             For i \leftarrow 1 to |V|:
 8
                   For j \leftarrow 1 to |V|:
 9
                         // if new path is shorter, reduce distance
10
                         \mathbf{If} \operatorname{dist}[i,j] > \operatorname{dist}[i,k] + \operatorname{dist}[k,j] :
11
                                \operatorname{dist}[i,j] \leftarrow \operatorname{dist}[i,k] + \operatorname{dist}[k,j]
12
13
14 Return dist
```

Alignment

indent-guides: 1pt + black block-align: bottom + right

```
Floyd-Warshall (V, E, w):
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2
    For (u, v) in E:
           \mathrm{dist}[u,v] \leftarrow w(u,v)
 3
                                                                               // edge weights
     For v in V:
 4
           \mathrm{dist}[v,v] \leftarrow 0
 5
                                                                               // base case
 6
 7
     For k \leftarrow 1 to |V|:
           For i \leftarrow 1 to |V|:
 8
                 For j \leftarrow 1 to |V|:
 9
                       // if new path is shorter, reduce distance
10
                       If dist[i, j] > dist[i, k] + dist[k, j]:
11
                             \mathrm{dist}[i,j] \leftarrow \mathrm{dist}[i,k] + \mathrm{dist}[k,j]
12
13
14 Return dist
```

```
1 def floyd_warshall(G):
2
     # let G be an adjacency matrix
     dist = G
3
4
     for k in range(len(G)):
5
       for i in range(len(G)):
6
         for j in range(len(G)):
7
          if dist[i][j] > dist[i][k] + dist[k][j]:
             dist[i][j] = dist[i][k] + dist[k][j]
10
     return dist
11
12
```

Breakable

indent-guides: 1pt + black breakable: true

```
\underline{\text{Floyd-Warshall}}(V,\,E,\,w)\text{:}
 1 Let \operatorname{dist}[u,v] \leftarrow \infty for u,v in V
 2 For (u, v) in E:
            \mathrm{dist}[u,v] \leftarrow w(u,v)
  3
                                                                                     // edge weights
  4 For v in V:
            \mathrm{dist}[v,v] \leftarrow 0
  5
                                                                                     // base case
  6
 7 For k \leftarrow 1 to |V|:
            For i \leftarrow 1 to |V|:
  8
                  For j \leftarrow 1 to |V|:
  9
```

Broken indent guides with small inset

row-gutter: 15pt inset: 3pt

 $indent-guides: \ \ \, 1pt+black$

breakable: true

```
      FLOYD-WARSHALL(V, E, w):

      1
      Let dist[u, v] \leftarrow \infty for u, v in V

      2
      For (u, v) in E:

      3
      dist[u, v] \leftarrow w(u, v)
      // edge weights

      4
      For v in V:

      5
      dist[v, v] \leftarrow 0
      // base case

      6
      7
      For k \leftarrow 1 to |V|:

      8
      For i \leftarrow 1 to |V|:

      9
      For j \leftarrow 1 to |V|:
```

```
1 def floyd_warshall(G):
2  # let G be an adjacency matrix
3  dist = G
4
5  for k in range(len(G)):
6   for i in range(len(G)):
7  for j in range(len(G)):
```