### 1.3 DIJKSTRA'S Two-STACK Algorithm


click to begin demo

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.
infix expression value stack operator stack
(fully parenthesized)


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
\begin{array}{cc}
\mathbf{1} & + \\
\text { value stack } & \text { operator stack }
\end{array}
$$



Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

2
$1+$
value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

2
1 +
value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

| $\mathbf{2}$ | + |
| :---: | :---: |
| $\mathbf{1}$ | + |
| value stack | operator stack |

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

| $\mathbf{2}$ | + |
| :---: | :---: |
| $\mathbf{1}$ | + |
| value stack | operator stack |

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

| $\mathbf{2}$ | + |
| :---: | :---: |
| $\mathbf{1}$ | + |
| value stack | operator stack |

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

3


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
3+2
$$



Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

1 +
value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

5
$1+$
value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

```
value stack operator stack
```

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

```
value stack operator stack
```

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

4

5
$1+$
value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

4


```
5 ) ()
```

$\uparrow$

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


$\uparrow$

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.
*
*
$+$
value stack
operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

| $\mathbf{5}$ |  |
| :---: | :---: |
| $\mathbf{4}$ |  |
| $\mathbf{5}$ |  |
| $\mathbf{1}$ |  |
| value stack | operator stack |

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.
$5 * 4=20$


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
20 * 5
$$



Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
20 * 5=100
$$



Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.


Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

```
            100
            1 +
value stack operator stack
```

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
100+1
$$

value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

$$
100+1=101
$$

value stack operator stack

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.

Dijkstra's two-stack algorithm

Value: push onto the value stack.
Operator: push onto the operator stack.
Left parenthesis: ignore.
Right parenthesis: pop operator and two values; push the result of applying that operator to those values onto the operand stack.
result

