Algorithms

Robert Sedgewick \| Kevin Wayne

### 1.3 Dijkstra's 2-Stack Demo

http://algs4.cs.princeton.edu

## 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 value 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.
$1+$
value stack operator stack
$(2+3) *(4 * 5)$

## 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.
$1+$
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.
$1+$
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.
$1+$
value stack operator stack
$2+3) *(4 * 5)$
$\uparrow$

## 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.

2
$1+$
value stack operator stack
$+3) *(4 * 5)$

## 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.

2
$1+$
value stack operator stack
$+3) *(4 * 5)$
$\uparrow$

## 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.

| 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

3 ) * ( 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.

3

| 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.

3

| 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.

$$
3+2
$$

$1+$
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.

$$
3+2=5
$$

$1+$
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.

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.


## 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
$1+$
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 | $*$ |  |
| :--- | :--- | :--- |
| $\mathbf{5}$ |  | $*$ |
| $\mathbf{1}$ |  | + |

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.

| 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.

5

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.

5
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.

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.
$5 * 4=20$


## 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.

$1+$
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.

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.

$$
20 * 5
$$

$1+$
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.

$$
20 * 5=100
$$

$1+$
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.

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.
$1+$
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.

101
result

```
(1+ + ( 2 + 3 ) * ( 4 * 5 ) ) )
```

