

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:53:49

JavaScript

```
class Stack {
  constructor() {
    this.stack = [];
  }
  push(v) {
    this.stack.push(v);
  }
  pop() {
    return this.stack.pop();
  }
  empty() {
    return this.stack.length === 0;
  }
}
```

Examples

```
adj_list = {
  1: [2, 3],
  2: [1, 3],
  3: [1, 2]
}
start = 1
result = []
```

And I'm going to add an empty()

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:53:29

JavaScript

```
class Stack {
  constructor() {
    this.stack = [];
  }
  push(v) {
    this.stack.push(v);
  }
  pop() {
    return this.stack.pop();
  }
  empty() {
    return this.stack.length === 0;
  }
}
```

Examples

```
adj_list = {
  1: [2, 3],
  2: [1, 3],
  3: [1, 2]
}
start = 1
result = []
```

And for empty, I'm going to return

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:53:00

JavaScript

```
main.js
4
5
6   push(v) {
7     this.stack.push(v);
8   }
9
10  pop() {
11    return this.stack.pop();
12  }
13
14  empty() {
15    return this.stack.length === 0;
16  }
17 }
18
19 function solution(adj_list, start) {
20   // Initialize visited set
21   let visited = new Set();
22   // Make a new stack
23   let stack = new Stack();
24   // Add the start node to the stack
25   stack.push(start);
26   // While the stack isn't empty
27   while (!stack.empty()) {
28     // Get the next node to visit
29     let node = stack.pop();
30     // If it is visited
31     if (visited.has(node)) continue;
32     // Add node to visited set
33     visited.add(node);
34     // Add all unvisited neighbors to the stack
35     for (let neighbor of adj_list[node]) {
36       if (!visited.has(neighbor)) {
37         stack.push(neighbor);
38       }
39     }
40   }
41   return Array.from(visited).sort((a, b) => a - b);
42 }
```

Examples

```
adj_list: [
  [1],
  [2, 3],
  [4],
  [1],
  [5]
]

(0) -> (1) -> (3)
      |
      v
(4) -> (2)

start: 0
result: [0, 1, 2, 4, 3]
```

We're going to initialize the visited

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:52:04

JavaScript

```
main.js
14   empty() {
15     return this.stack.length === 0;
16   }
17 }
18
19 function solution(adj_list, start) {
20   // Initialize visited set to empty
21   let visited = new Set();
22   // Make a new stack
23   let stack = new Stack();
24   // Add the start node to the stack
25   stack.push(start);
26   // While the stack isn't empty
27   while (!stack.empty()) {
28     // Get the next node to visit
29     let node = stack.pop();
30     // If it is visited
31     if (visited.has(node)) continue;
32     // Add node to visited set
33     visited.add(node);
34     // Add all unvisited neighbors to the stack
35     for (let neighbor of adj_list[node]) {
36       if (!visited.has(neighbor)) {
37         stack.push(neighbor);
38       }
39     }
40   }
41   return Array.from(visited).sort((a, b) => a - b);
42 }
```

Examples

```
adj_list: [
  [1],
  [2, 3],
  [4],
  [1],
  [5]
]

(0) -> (1) -> (3)
      |
      v
(4) -> (2)

start: 0
result: [0, 1, 2, 4, 3]
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

BACK TO QUESTIONS Question 1 of 3 8,759:52:04

Java: A java.util.Stack with .push() , .pop() , and .empty() .

Examples

```
adj_list: [
  [1],
  [2,3],
  [4],
  [1],
  [5]
]

(0) -> (1) -> (3)
      |
      v
(4) <- (2)

start: 0
result: [0, 1, 2, 4, 3]
```

main.js

```
13
14
15 empty() {
16   return this.stack.length === 0;
17 }
18
19 function solution(adj_list, start) {
20   // Initialize visited set to empty
21
22   // Make a new stack
23
24   // Add the start node to the stack
25
26   // While the stack isn't empty
27   // Get the next node to visit
28
29   // If it is visited
30   // continue
31 }
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

BACK TO QUESTIONS Question 1 of 3 8,759:48:59

Java: A java.util.Stack with .push() , .pop() , and .empty() .

Examples

```
adj_list: [
  [1],
  [2,3],
  [4],
  [1],
  [5]
]

(0) -> (1) -> (3)
      |
      v
(4) <- (2)

start: 0
result: [0, 1, 2, 4, 3]
```

main.js

```
16
17
18 function solution(adj_list, start) {
19   // Initialize result array to empty
20   const result = [];
21
22   // Initialize visited set to empty
23   const visited = {};
24
25   // Make a new stack
26   const to_visit = new Stack();
27
28   // Add the start node to the stack
29   to_visit.push(start);
30
31   // While the stack isn't empty
32   // Get the next node to visit
33
34   // If it is visited
35   // continue
36 }
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:48:35

```
Java: A java.util.Stack with .push(), .pop(), and .empty().
```

Examples

```
adj_list: {
  (1): [
    (2,3),
    (4),
    (1),
    (5)
  ]
}
(0) -> (1) -> (3)
      |
      v
(4) <- (2)
start: 0
result: [0, 1, 2, 4, 3]
```

```
main.js
10
11
12 function solution(adj_list, start) {
13   // Initialize result array to empty
14   const result = [];
15
16   // Initialize visited set to empty
17   const visited = {};
18
19   // Make a new stack
20   const to_visit = new Stack();
21
22   // Add the start node to the stack
23   to_visit.push(start);
24
25   // While the stack isn't empty
26   while (!to_visit.empty()) {
27     // Get the next node to visit
28     const n = to_visit.pop();
29
30     // If it is visited
31     if (visited[n]) continue;
32
33     // Add it to the result
34     result.push(n);
35     visited[n] = true;
36
37     // Add its neighbors to the stack
38     for (const neighbor of adj_list[n]) {
39       to_visit.push(neighbor);
40     }
41   }
42
43   return result;
44 }
```

So I'll call this n and let's just

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:48:10

```
Java: A java.util.Stack with .push(), .pop(), and .empty().
```

Examples

```
adj_list: {
  (1): [
    (2,3),
    (4),
    (1),
    (5)
  ]
}
(0) -> (1) -> (3)
      |
      v
(4) <- (2)
start: 0
result: [0, 1, 2, 4, 3]
```

```
main.js
21
22 const result = [];
23
24 // Initialize visited set to empty
25 const visited = {};
26
27 // Make a new stack
28 const to_visit = new Stack();
29
30 // Add the start node to the stack
31 to_visit.push(start);
32
33 // While the stack isn't empty
34 while (!to_visit.empty()) {
35   // Get the next node to visit
36   const n = to_visit.pop();
37
38   // If it is visited
39   if (visited[n]) continue;
40
41   // Add it to the result
42   result.push(n);
43   visited[n] = true;
44
45   // Add its neighbors to the stack
46   for (const neighbor of adj_list[n]) {
47     to_visit.push(neighbor);
48   }
49 }
```

Let's see, if n is in visited, we've

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:47:33

JavaScript

```
main.js
const to_visit = new Stack();
// Add the start node to the stack
to_visit.push(start);

// While the stack isn't empty
while (!to_visit.empty()) {
  // Get the next node to visit
  const n = to_visit.pop();

  // If it is visited, skip it
  if (n in visited) continue;

  // Mark as visited
  visited[n] = true;

  // Add to result
```

Examples

```
adj_list: {
  [1]: [2, 3],
  [4]: [1],
  [5]: [6]
}

(0) -> (1) -> (3)
+
1
+
(4) -> (2)

start: 0
result: [0, 1, 2, 4, 3]
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:47:30

JavaScript

```
main.js
constructor() {
  this.stack = [];
}

push(v) {
  this.stack.push(v);
}

pop() {
  return this.stack.pop();
}

empty() {
  return this.stack.length === 0;
}

function solution(adj_list, start) {
  // Initialize result array to empty
```

BloomTech Portal

Web ACS Web Labs Web

Home Classes Course Materials Gradebook Sprint Challenges GCA Book Meeting Support

Iterative Depth-First Search Implementation

```
graph LR
    subgraph Examples
        A["adj_list: [
            [1],
            [2,3],
            [4],
            [1],
            [0]
        ]"]
        B["(0) -> (1) -> (3)
            + 1
            | w
            (4) -> (2)"]
        C["start: 0
            result: [0, 1, 2, 4, 3]"]
    end
    subgraph mainjs
        D["// Mark as visited
            visited[n] = true;

            // Add to results
            result.push(n);

            // Go through all the neighbors
            const neighbors = adj_list[n];

            // Push each neighbor on the stack
            for (let neighbor of neighbors)
                to_visit.push(neighbor);

            // Return the result
            return result;"]
    end
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:45:55

```
Java: A java.util.Stack with .push(), .pop(), and .empty().
```

Examples

```
adj_list: {
  (1),
  (2,3),
  (4),
  (1),
  (0)
}

(0) -> (1) -> (3)
      |
      v
(4) -> (2)
```

start: 0
result: [0, 1, 2, 4, 3]

```
main.js
29 // Add the start node to the stack
30 to_visit.push(start);
31
32 // While the stack isn't empty
33 while (!to_visit.empty()) {
34   // Get the next node to visit
35   const n = to_visit.pop();
36
37   // If it is visited, skip it
38   if (n in visited)
39     continue
40
41   // Mark as visited
42   visited[n] = true;
43
44   // Add to results
45   result.push(n);
46
47   // Go through all the neighbors
```

portal.bloomtech.com/student/course-materials

BloomTech Portal

Web ACS Web Labs Web

Home
Classes
Course Materials
Gradebook
Sprint Challenges
GCA
Book Meeting
Support

Iterative Depth-First Search Implementation

Question 1 of 3 8,759:45:55

```
Java: A java.util.Stack with .push(), .pop(), and .empty().
```

Examples

```
adj_list: {
  (1),
  (2,3),
  (4),
  (1),
  (0)
}

(0) -> (1) -> (3)
      |
      v
(4) -> (2)
```

start: 0
result: [0, 1, 2, 4, 3]

```
main.js
29 // Add the start node to the stack
30 to_visit.push(start);
31
32 // While the stack isn't empty
33 while (!to_visit.empty()) {
34   // Get the next node to visit
35   const n = to_visit.pop();
36
37   // If it is visited, skip it
38   if (n in visited)
39     continue
40
41   // Mark as visited
42   visited[n] = true;
43
44   // Add to results
45   result.push(n);
46
47   // Go through all the neighbors
```

