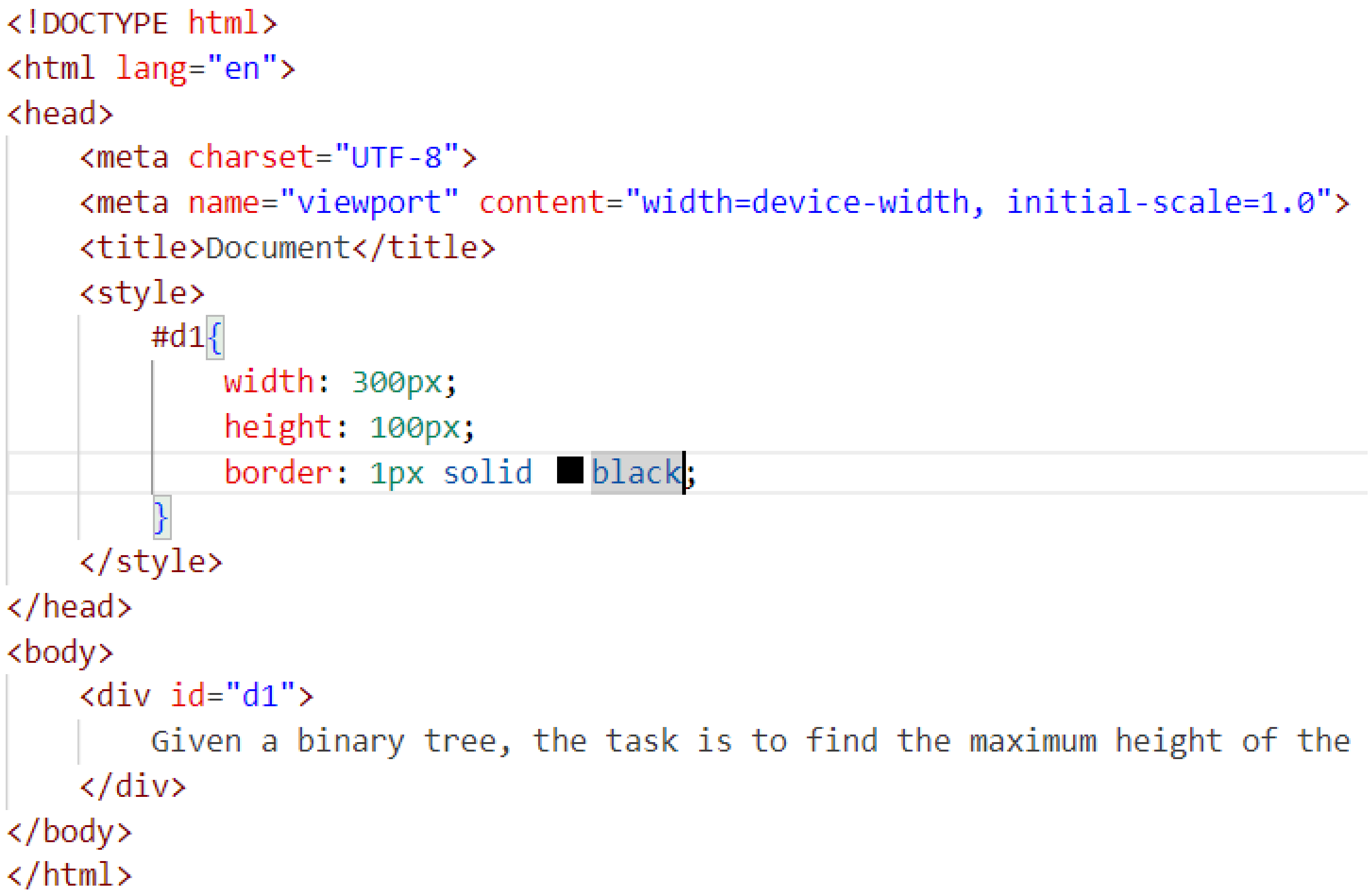
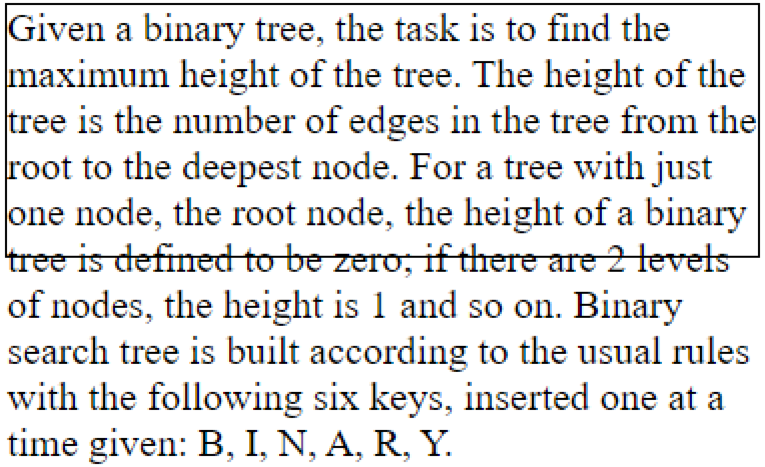
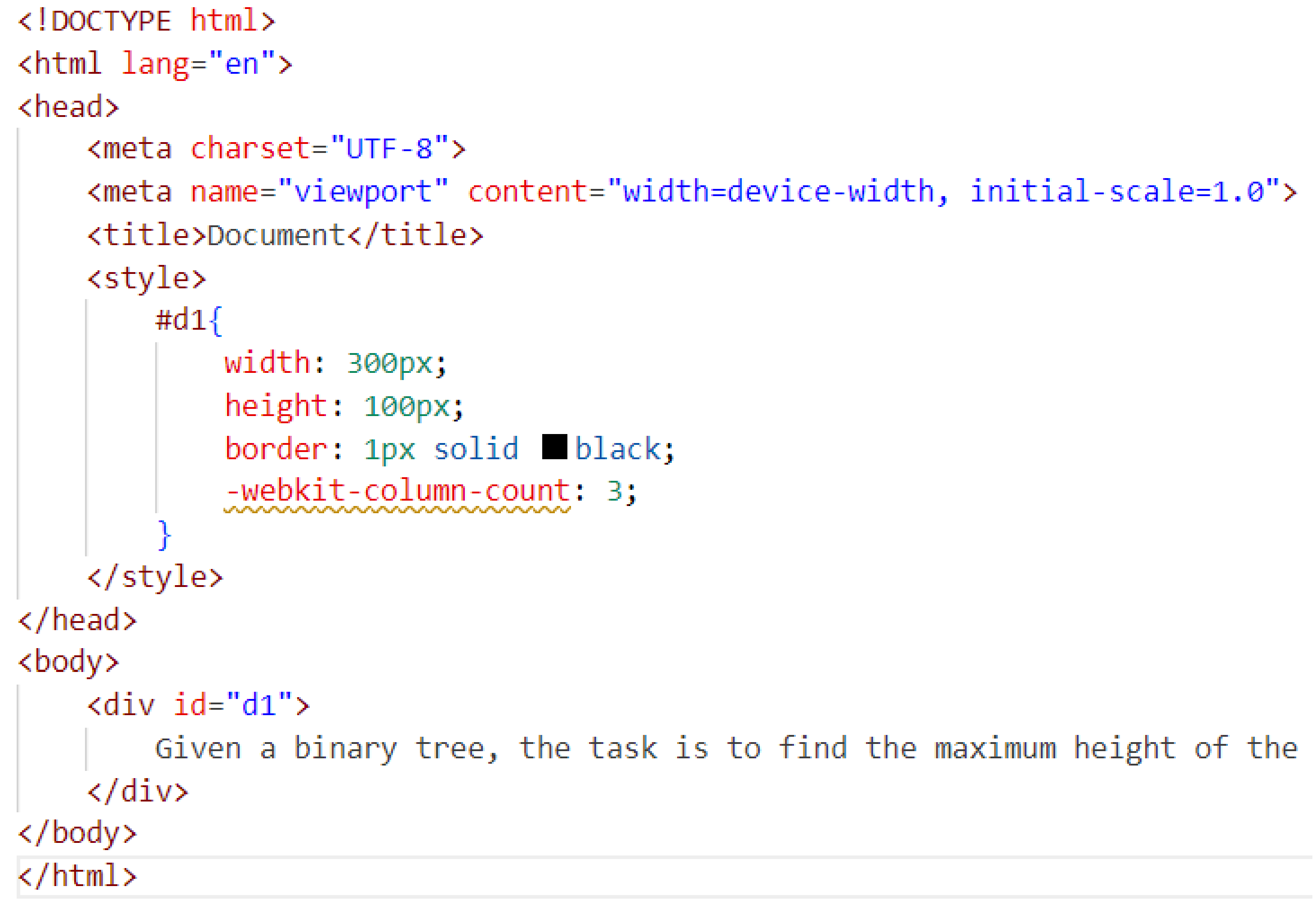
Column



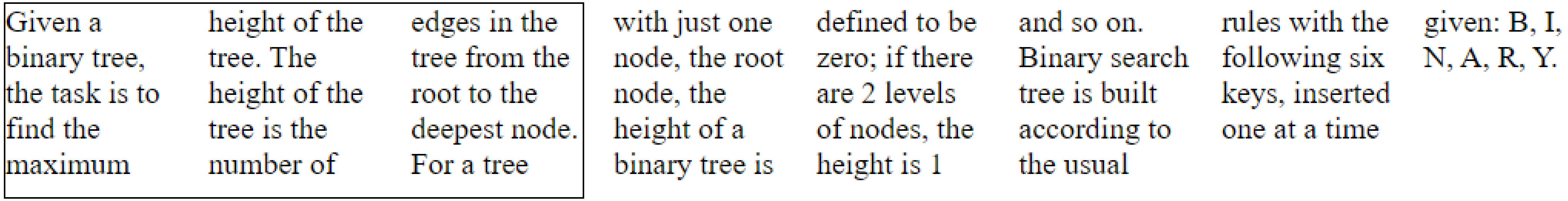
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■ black;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



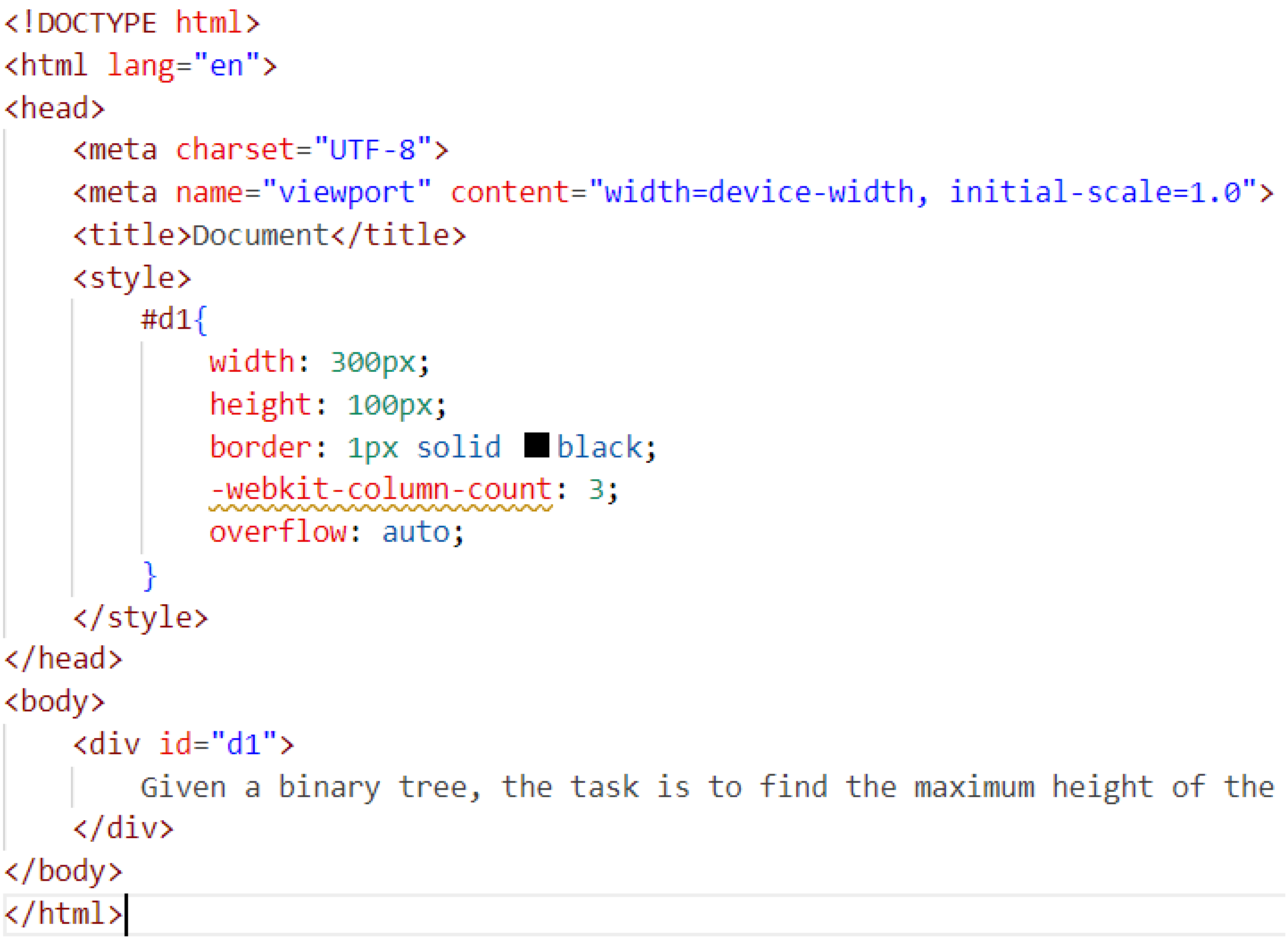
Given a binary tree, the task is to find the maximum height of the tree. The height of the tree is the number of edges in the tree from the root to the deepest node. For a tree with just one node, the root node, the height of a binary tree is defined to be zero; if there are 2 levels of nodes, the height is 1 and so on. Binary search tree is built according to the usual rules with the following six keys, inserted one at a time given: B, I, N, A, R, Y.



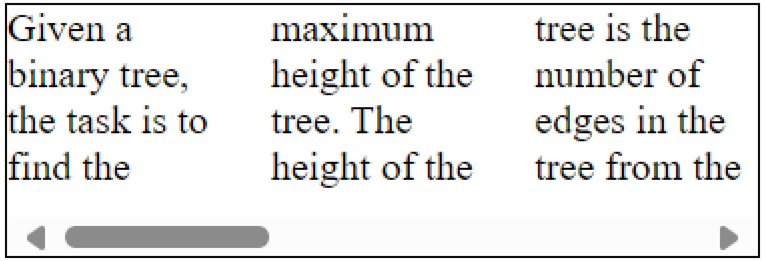
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid black;
            -webkit-column-count: 3;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



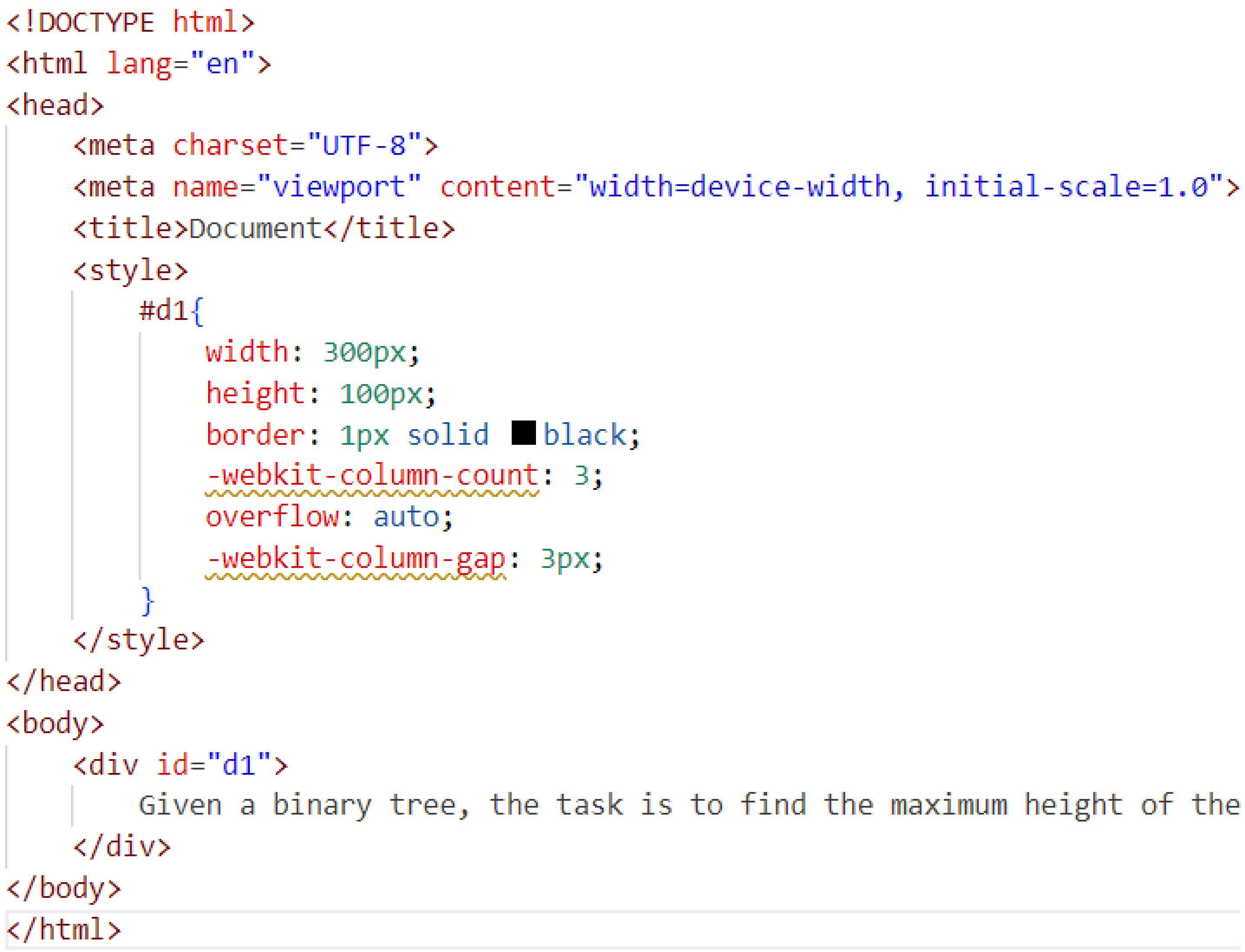
Given a binary tree, the task is to find the maximum height of the tree. The height of the tree is the number of edges in the tree from the root to the deepest node. For a tree with just one node, the root node, the height of a binary tree is defined to be zero; if there are 2 levels of nodes, the height is 1 and so on. Binary search tree is built according to the usual rules with the following six keys, inserted one at a time given: B, I, N, A, R, Y.



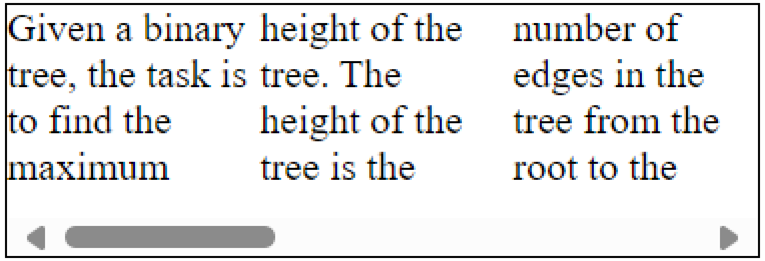
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■ black;
            -webkit-column-count: 3;
            overflow: auto;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



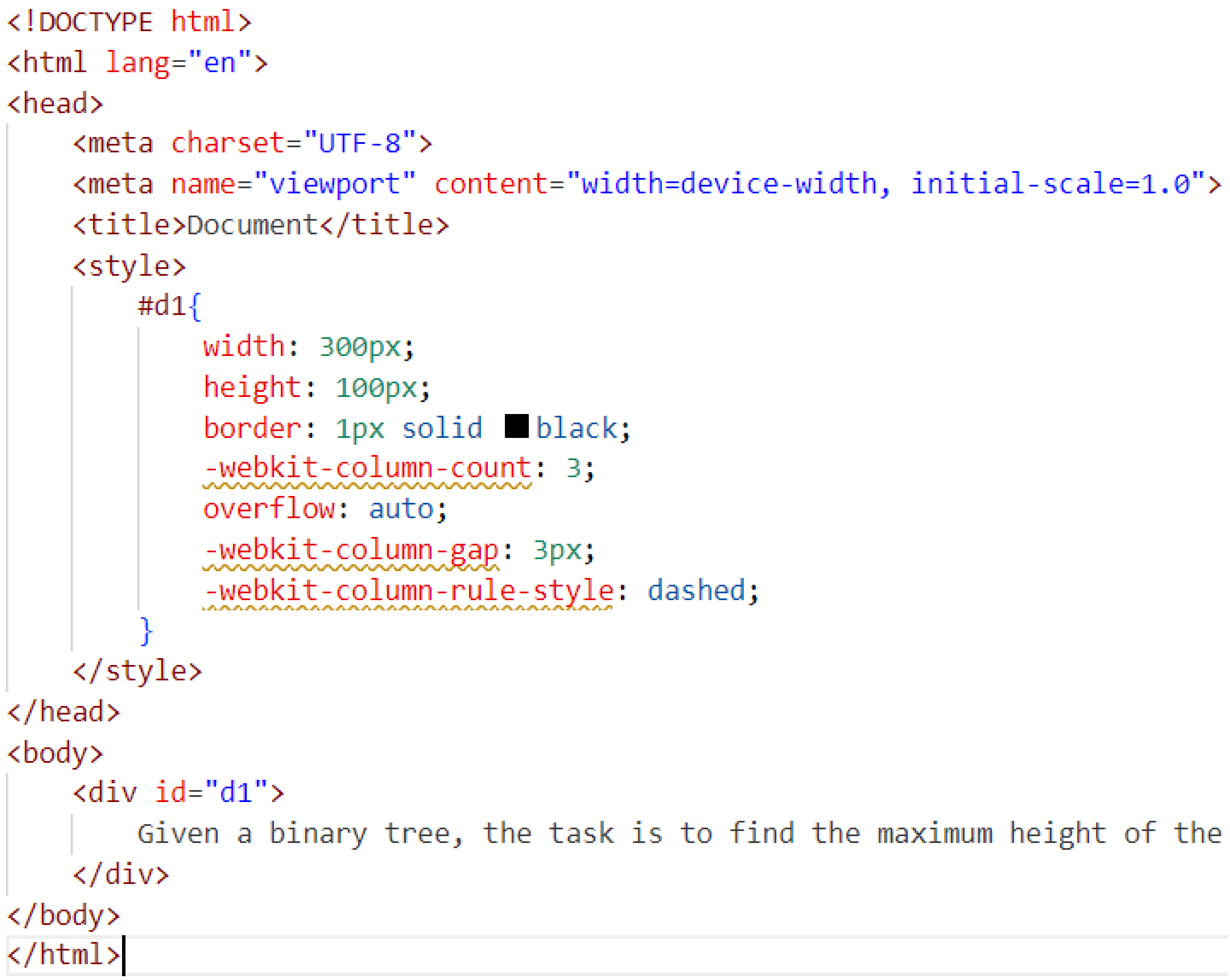
```
Given a            maximum           tree is the
binary tree,       height of the     number of
the task is to     tree. The         edges in the
find the           height of the     tree from the
```



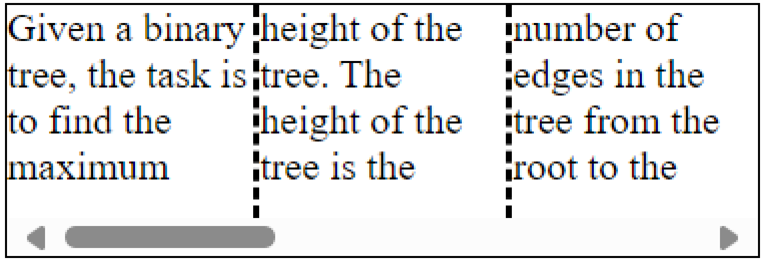
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 3px;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



```
Given a binary   height of the   number of
tree, the task is tree. The      edges in the
to find the      height of the   tree from the
maximum          tree is the     root to the
```



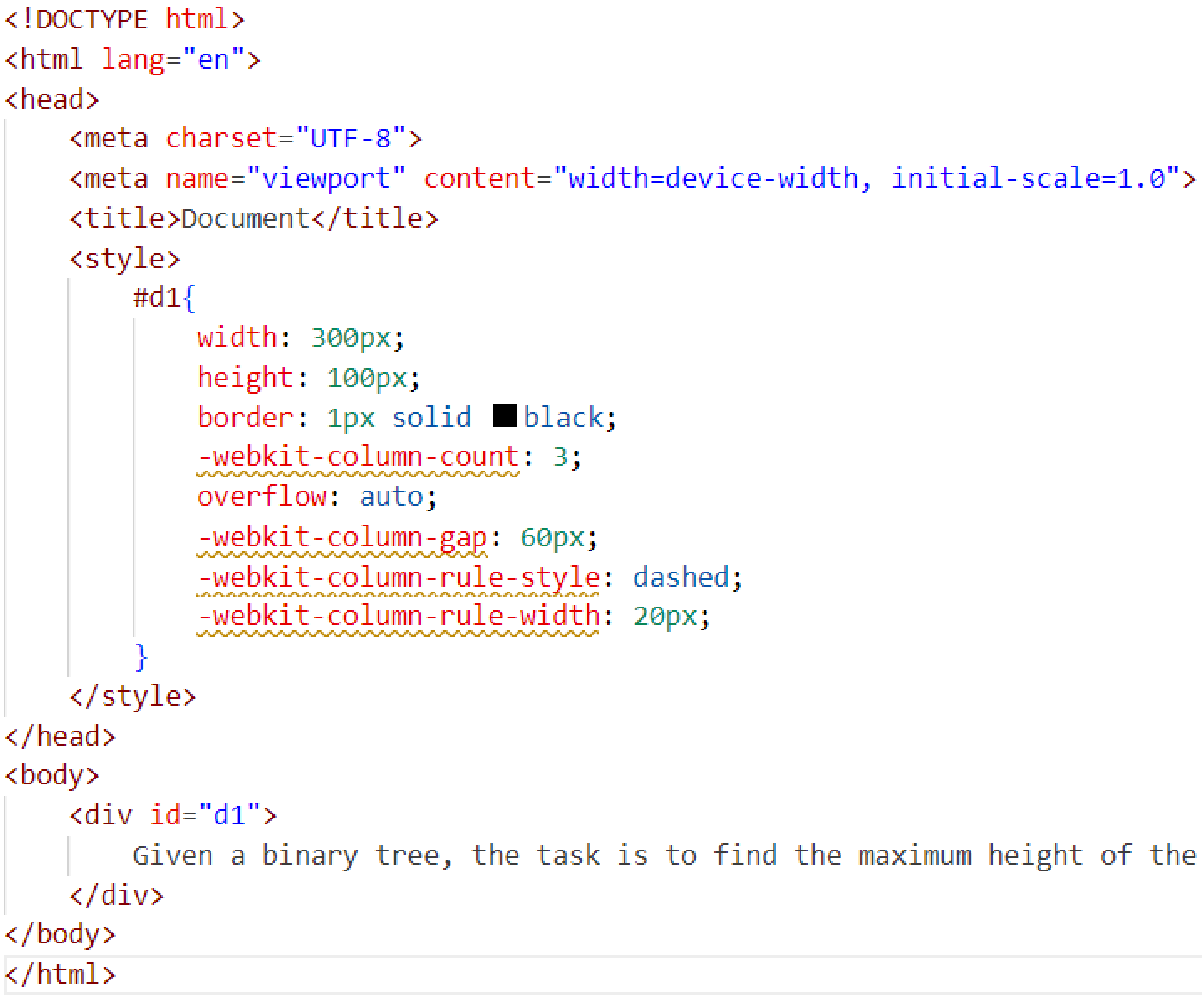
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 3px;
            -webkit-column-rule-style: dashed;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



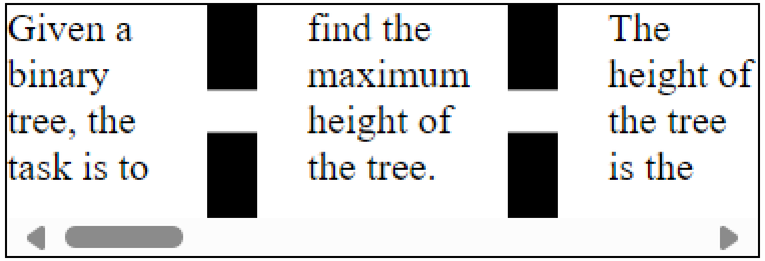
Given a binary tree, the task is to find the maximum

height of the tree. The height of the tree is the

number of edges in the tree from the root to the



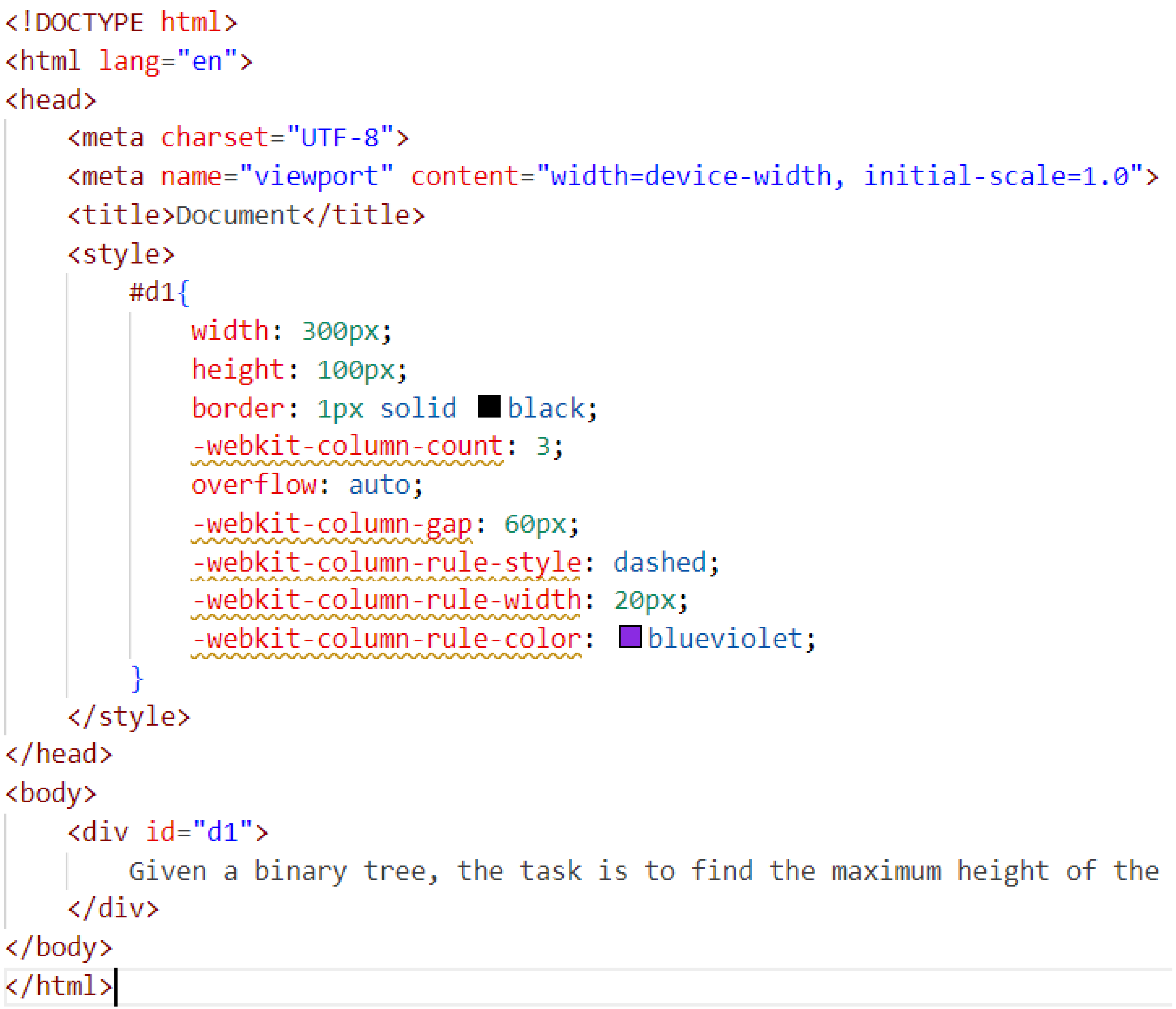
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 60px;
            -webkit-column-rule-style: dashed;
            -webkit-column-rule-width: 20px;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



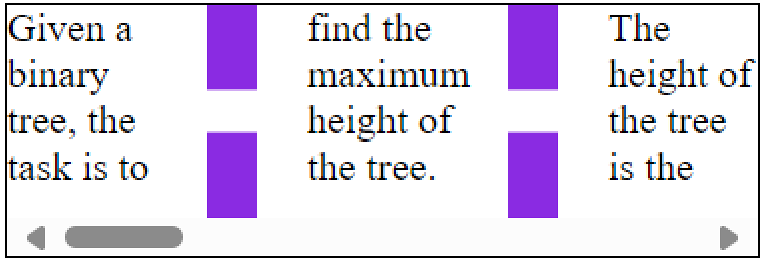
Given a binary tree, the task is to

find the maximum height of the tree.

The height of the tree is the



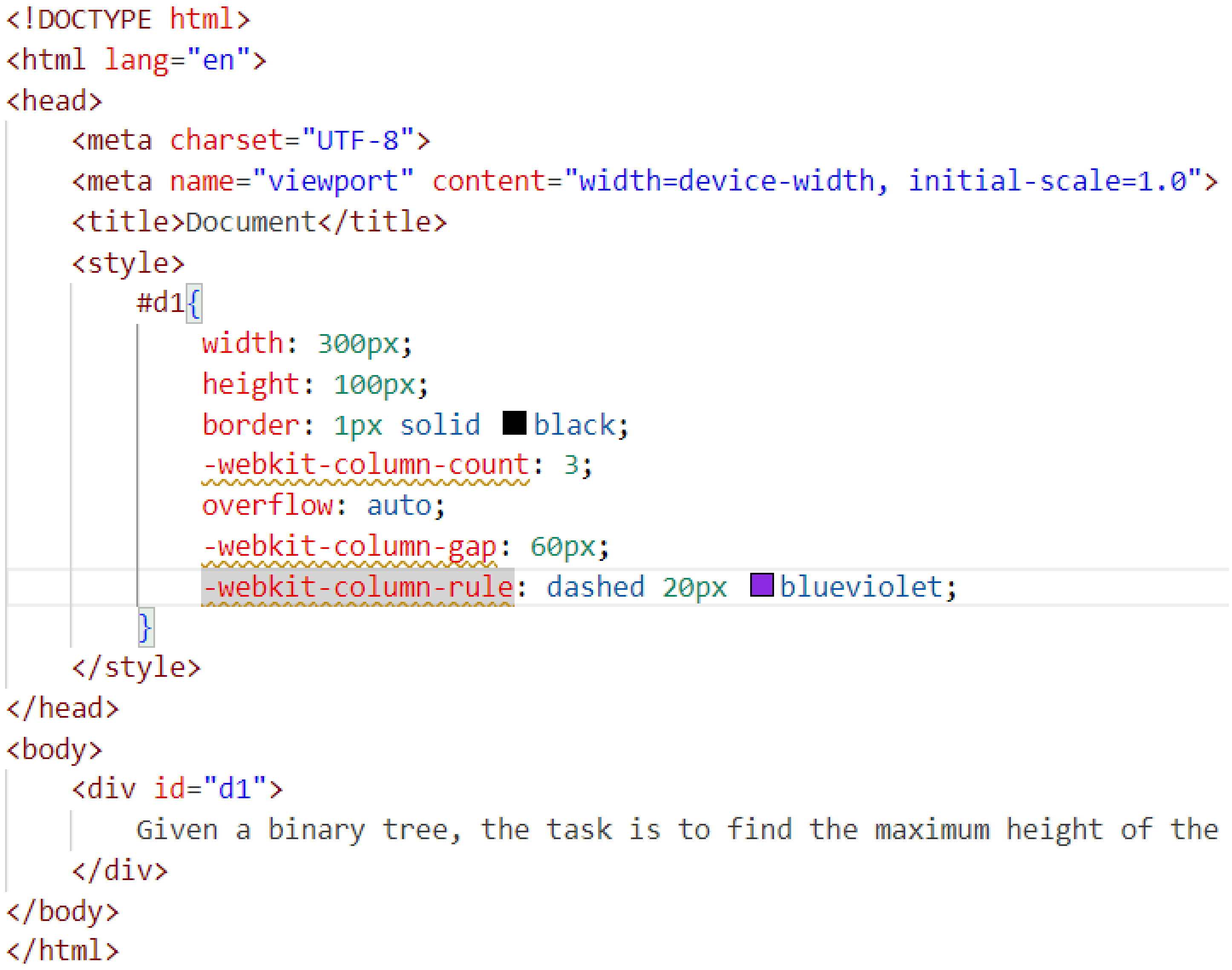
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 60px;
            -webkit-column-rule-style: dashed;
            -webkit-column-rule-width: 20px;
            -webkit-column-rule-color: 🟪blueviolet;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



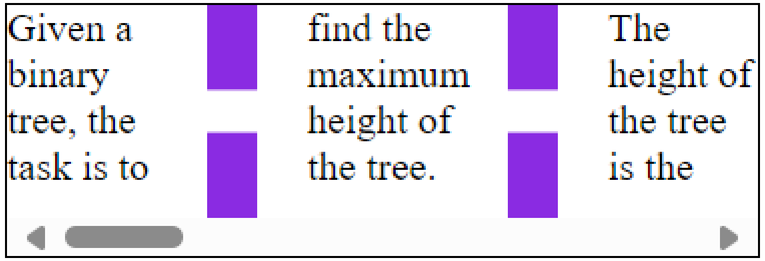
Given a binary tree, the task is to

find the maximum height of the tree.

The height of the tree is the



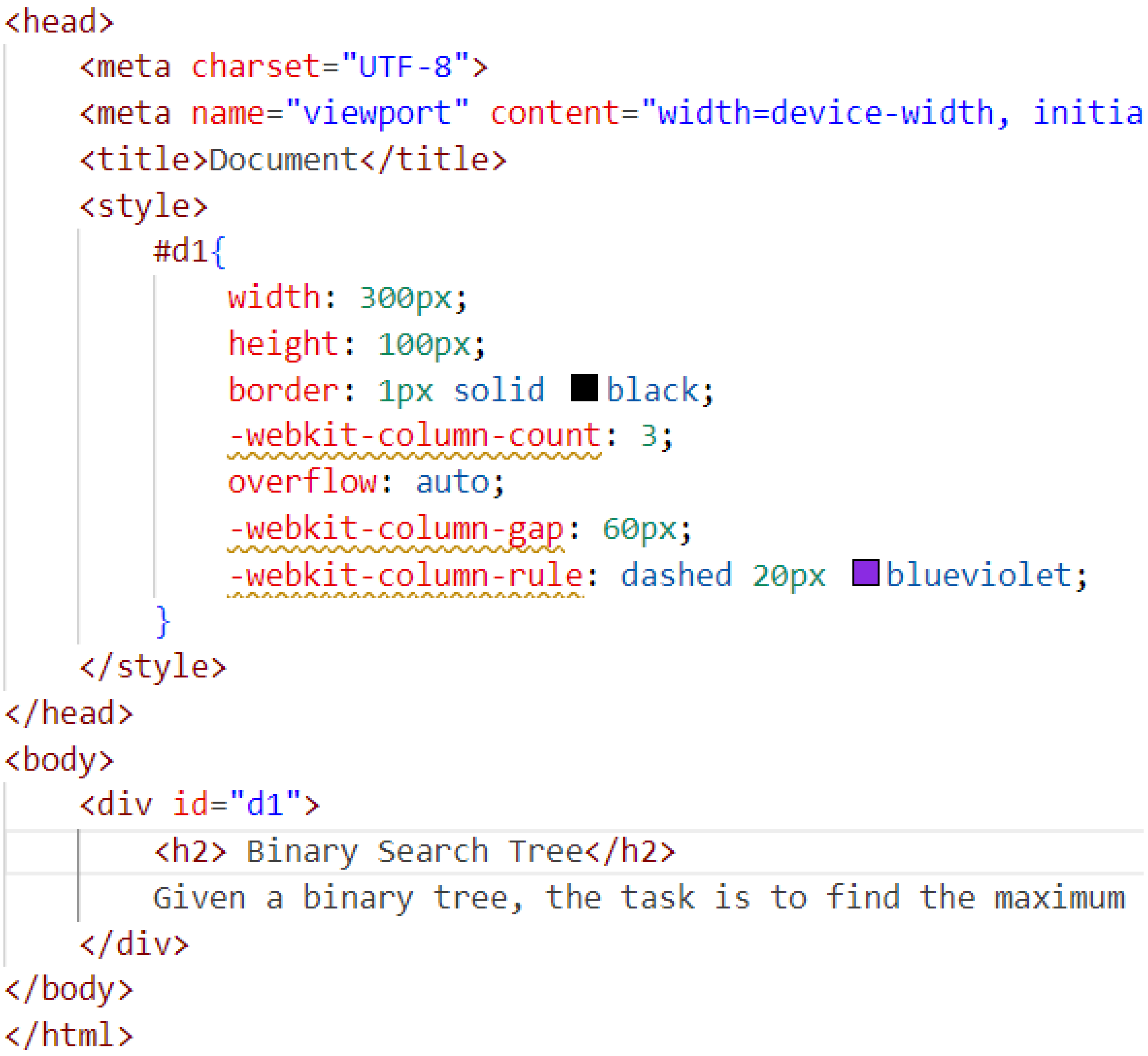
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 60px;
            -webkit-column-rule: dashed 20px blueviolet;
        }
    </style>
</head>
<body>
    <div id="d1">
        Given a binary tree, the task is to find the maximum height of the
    </div>
</body>
</html>
```



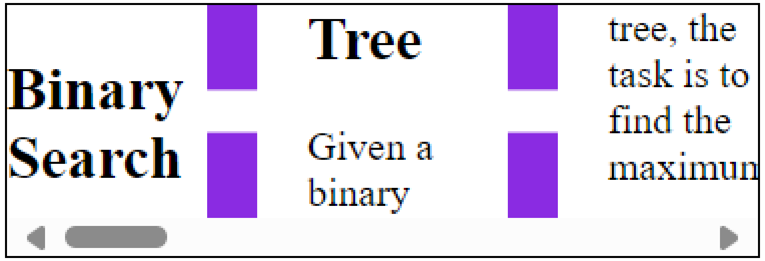
```
Given a
binary
tree, the
task is to

find the
maximum
height of
the tree.

The
height of
the tree
is the
```

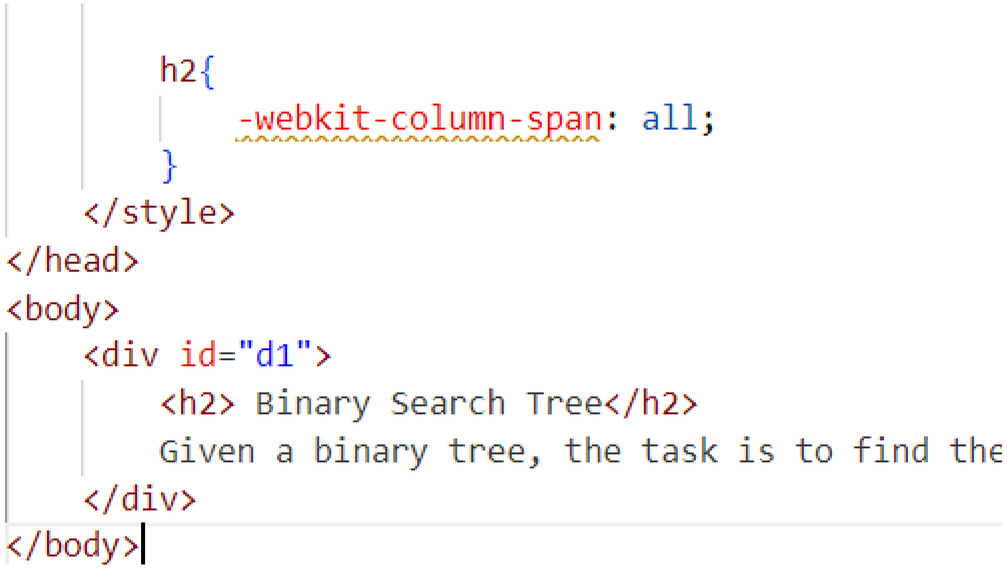


```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initia
    <title>Document</title>
    <style>
        #d1{
            width: 300px;
            height: 100px;
            border: 1px solid ■black;
            -webkit-column-count: 3;
            overflow: auto;
            -webkit-column-gap: 60px;
            -webkit-column-rule: dashed 20px 🟪blueviolet;
        }
    </style>
</head>
<body>
    <div id="d1">
        <h2> Binary Search Tree</h2>
        Given a binary tree, the task is to find the maximum
    </div>
</body>
</html>
```

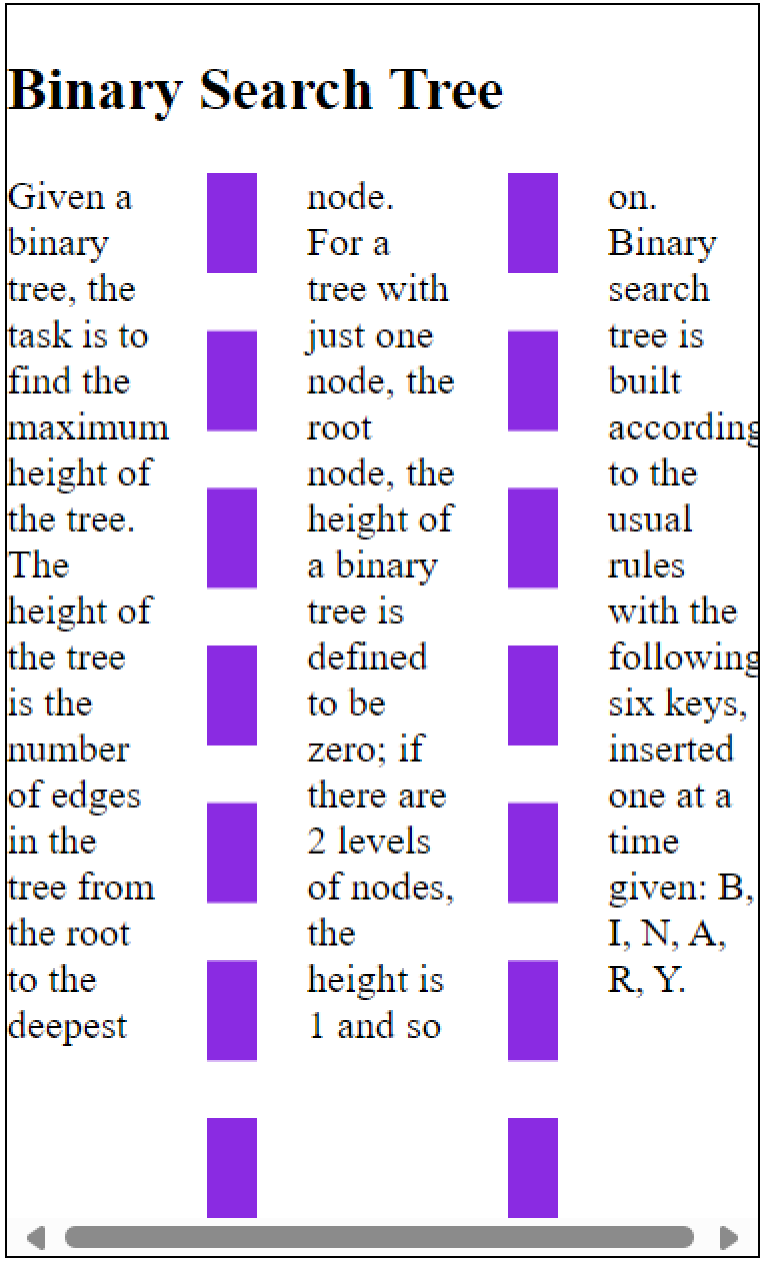


Binary Search **Tree**

Given a binary tree, the task is to find the maximun

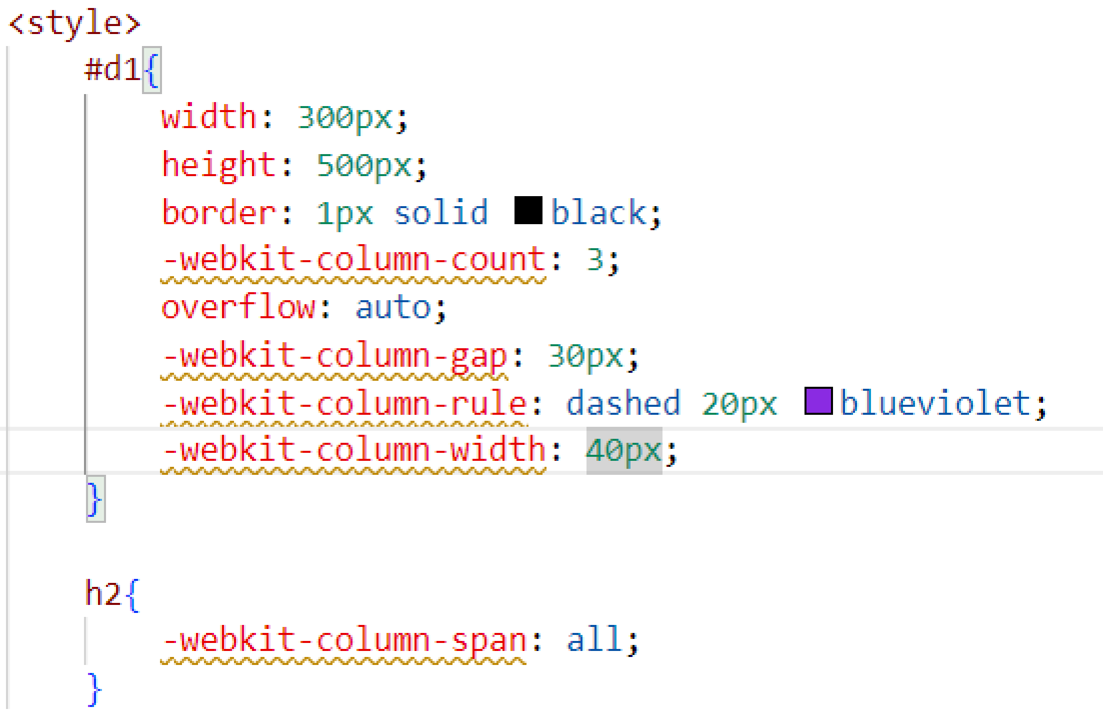


```
        h2{
            -webkit-column-span: all;
        }
    </style>
</head>
<body>
    <div id="d1">
        <h2> Binary Search Tree</h2>
        Given a binary tree, the task is to find the
    </div>
</body>
```



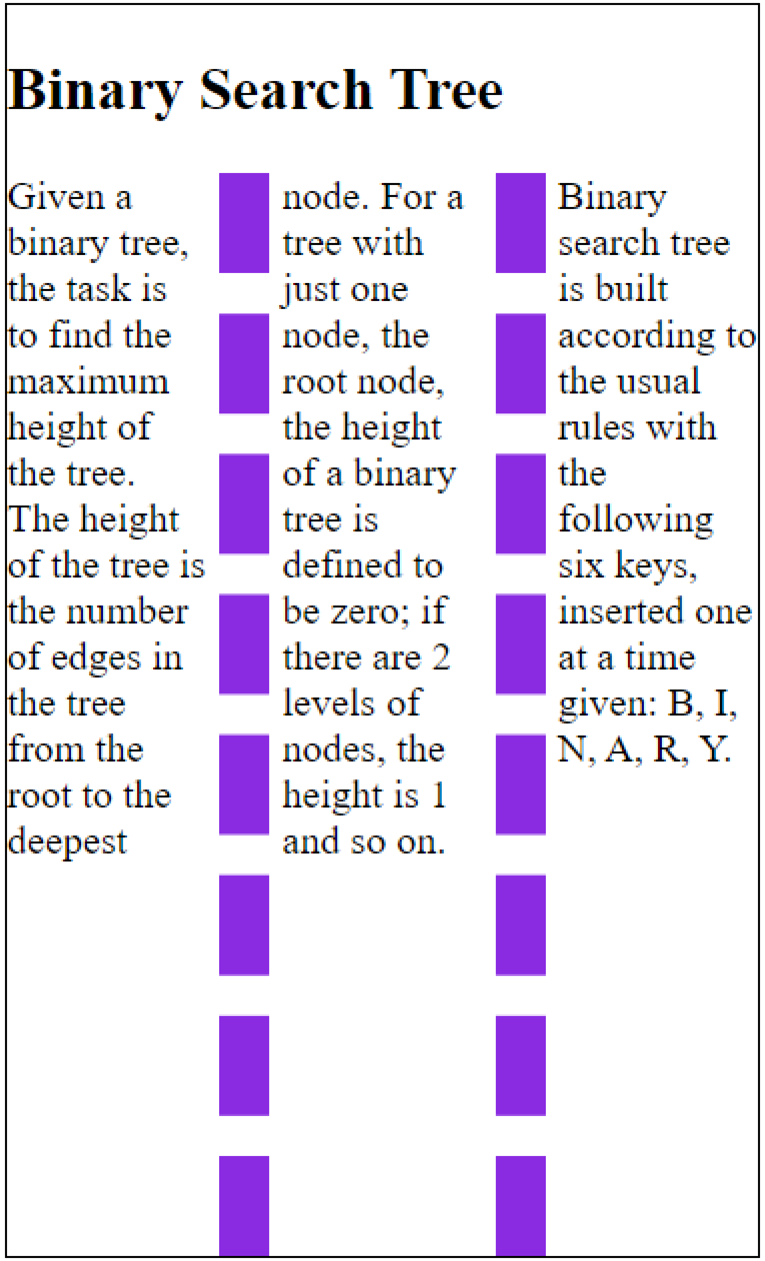
Binary Search Tree

Given a binary tree, the task is to find the maximum height of the tree. The height of the tree is the number of edges in the tree from the root to the deepest node. For a tree with just one node, the root node, the height of a binary tree is defined to be zero; if there are 2 levels of nodes, the height is 1 and so on. Binary search tree is built according to the usual rules with the following six keys, inserted one at a time given: B, I, N, A, R, Y.



```
<style>
    #d1{
        width: 300px;
        height: 500px;
        border: 1px solid black;
        -webkit-column-count: 3;
        overflow: auto;
        -webkit-column-gap: 30px;
        -webkit-column-rule: dashed 20px blueviolet;
        -webkit-column-width: 40px;
    }

    h2{
        -webkit-column-span: all;
    }
```



Binary Search Tree

Given a binary tree, the task is to find the maximum height of the tree. The height of the tree is the number of edges in the tree from the root to the deepest node. For a tree with just one node, the root node, the height of a binary tree is defined to be zero; if there are 2 levels of nodes, the height is 1 and so on. Binary search tree is built according to the usual rules with the following six keys, inserted one at a time given: B, I, N, A, R, Y.