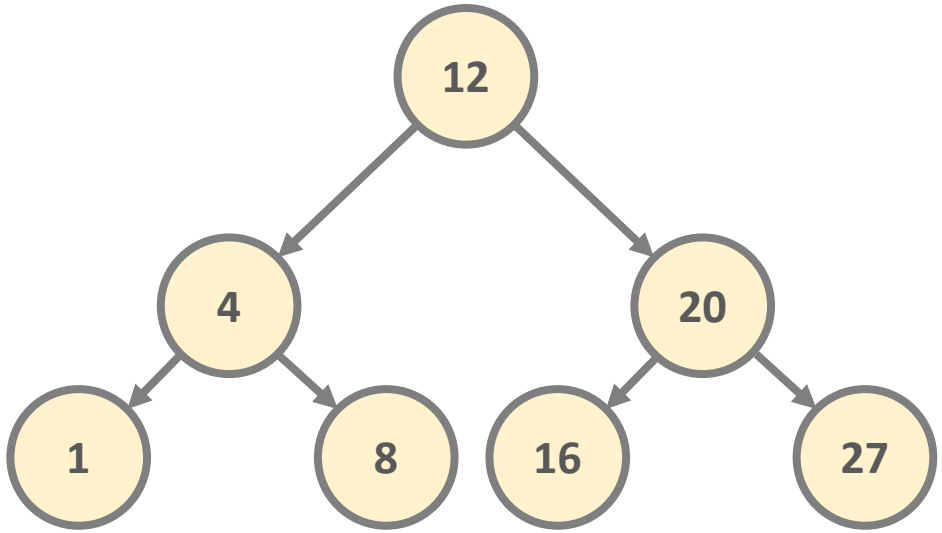
Stack Memory Visualization
(Algorithmic Problems)

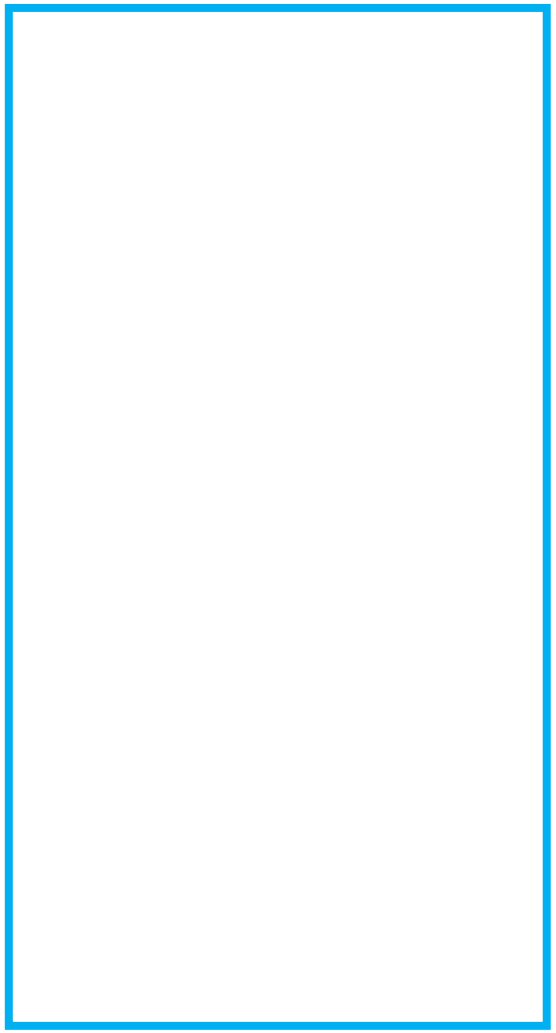
Binary Search Trees



max(node):

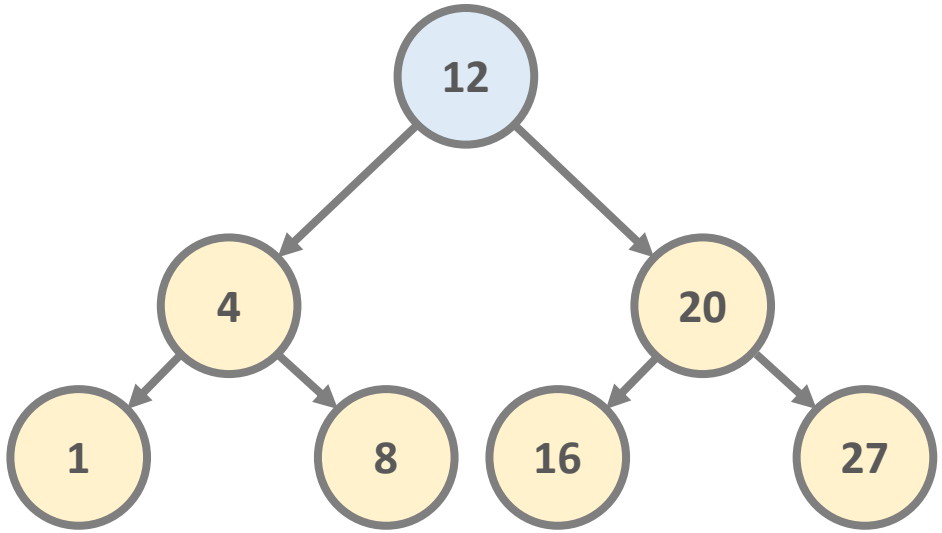
if node right child is not NULL:
return max(node right child)

return node value



STACK

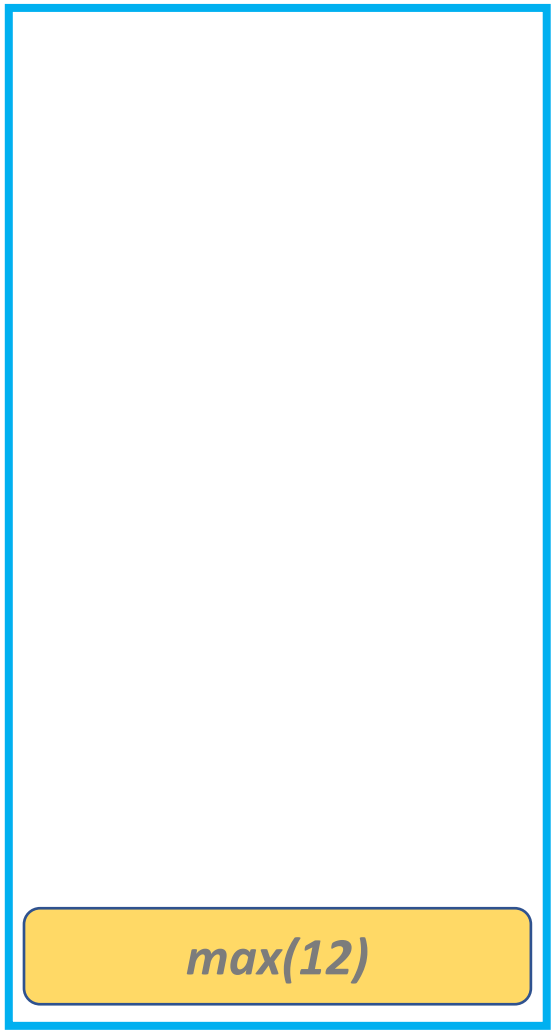
Binary Search Trees



max(node):

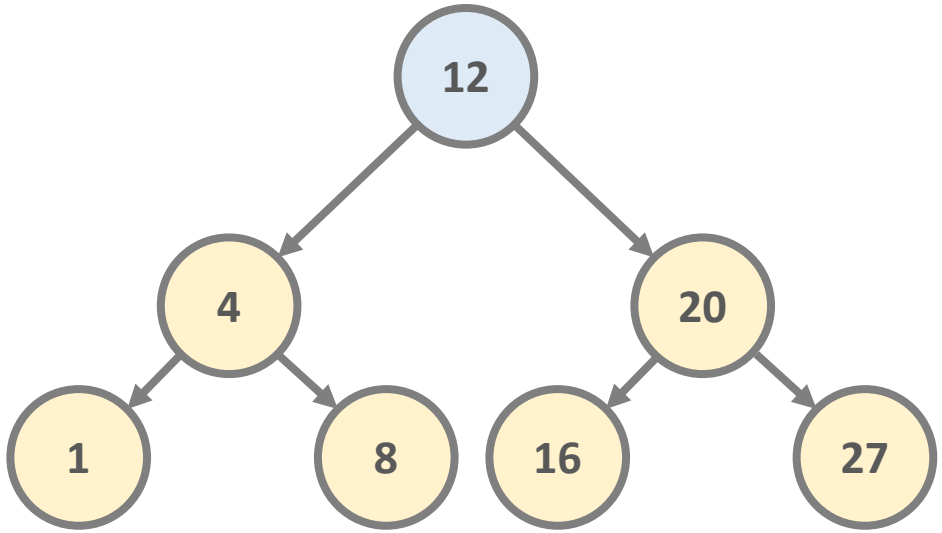
if node right child is not NULL:
return max(node right child)

return node value

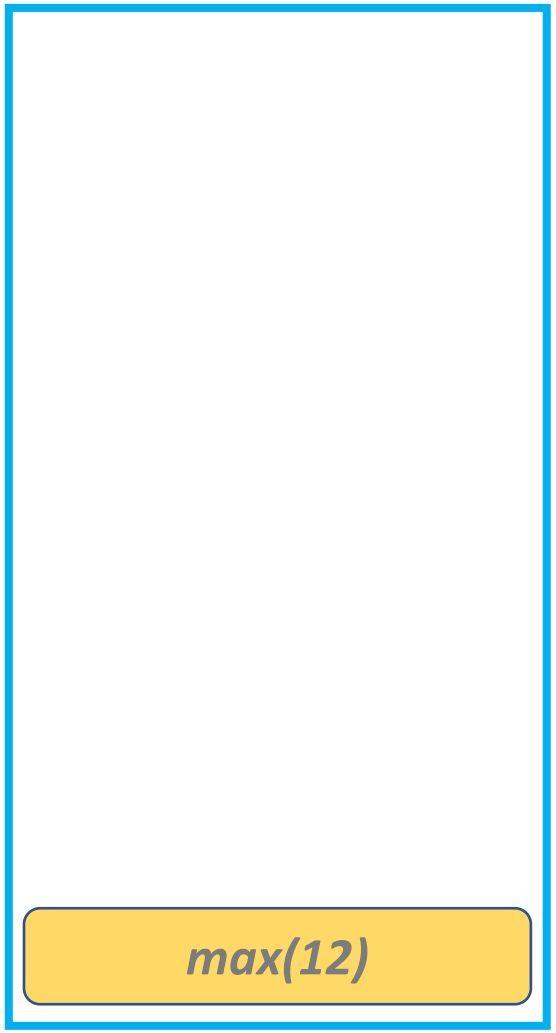


STACK

Binary Search Trees

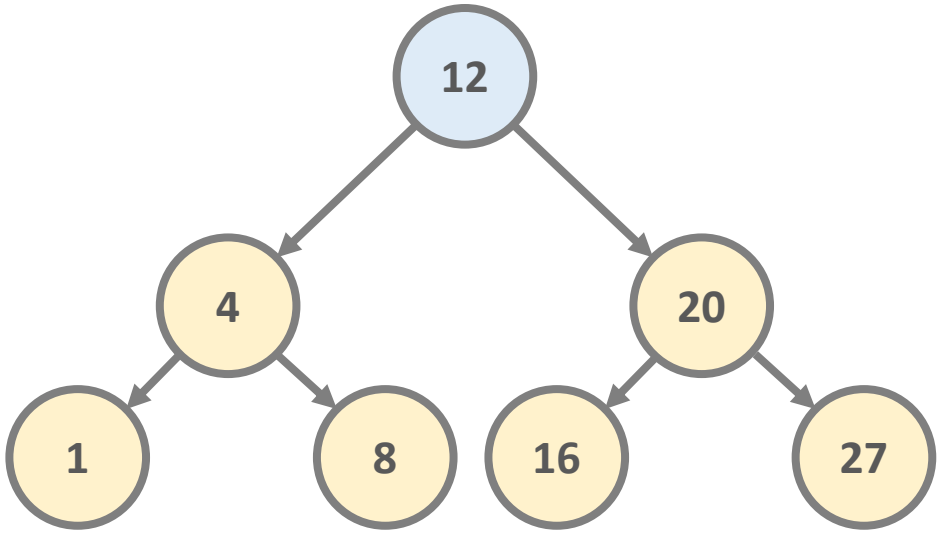



```
max(node):

    if node right child is not NULL:
        return max(node right child)

    return node value
```



STACK

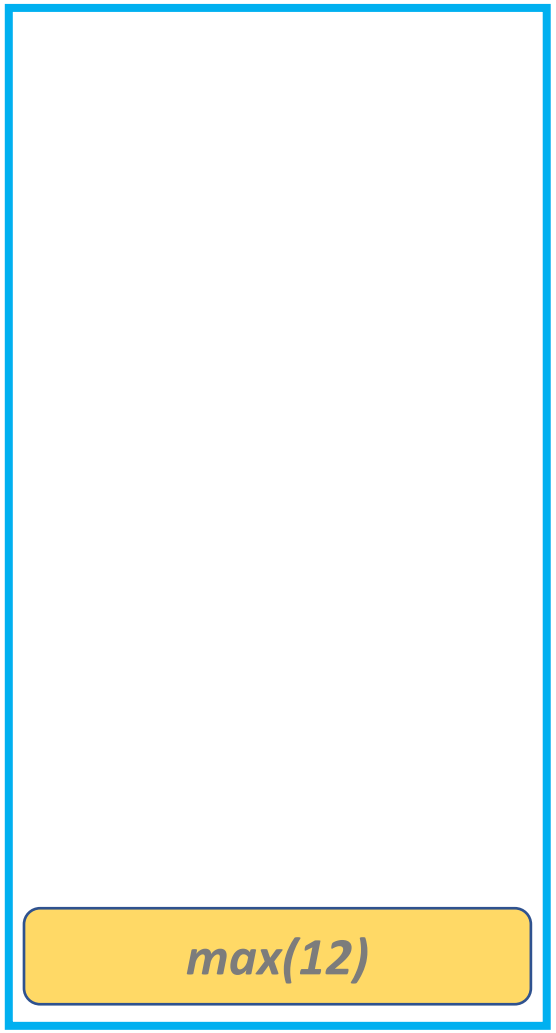
Binary Search Trees



max(node):

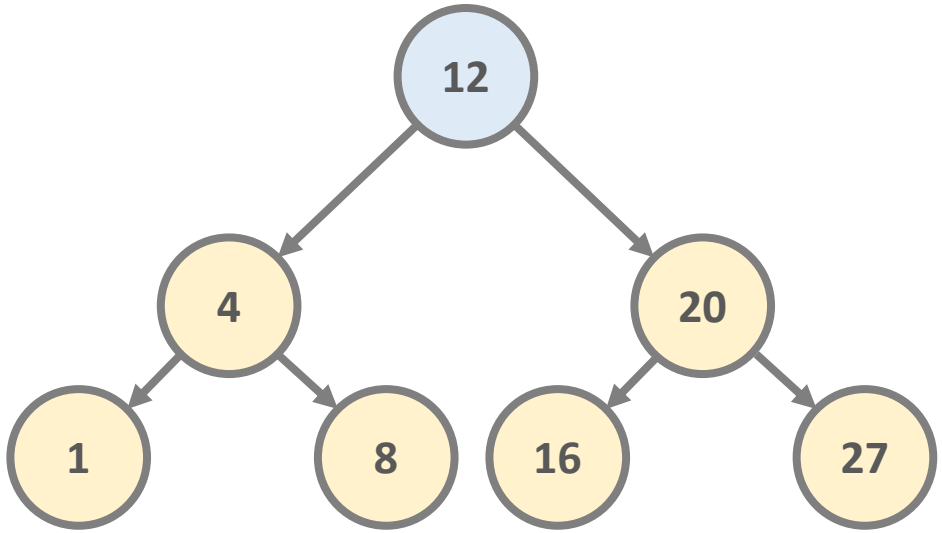
if node right child is not NULL:
return max(node right child)

return node value



STACK

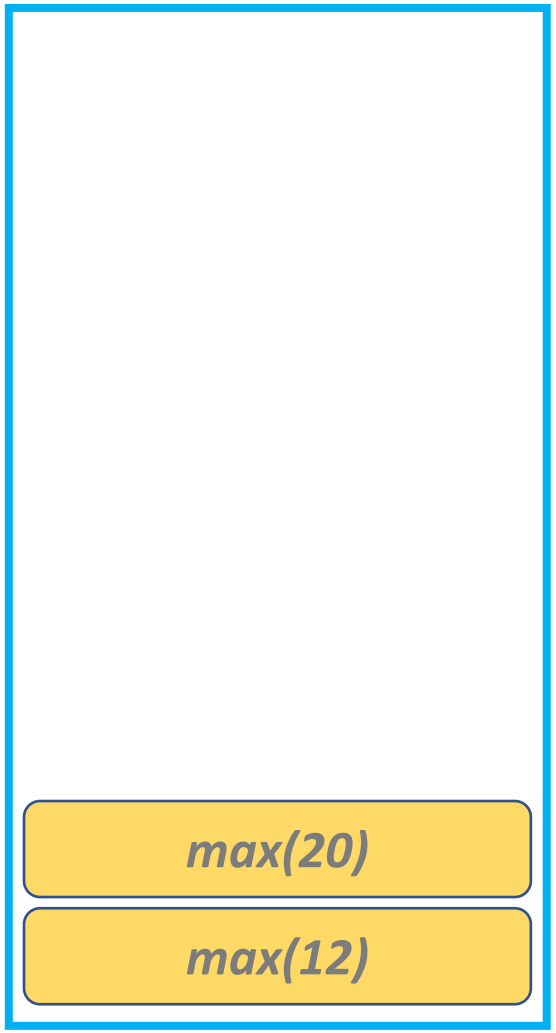
Binary Search Trees



max(node):

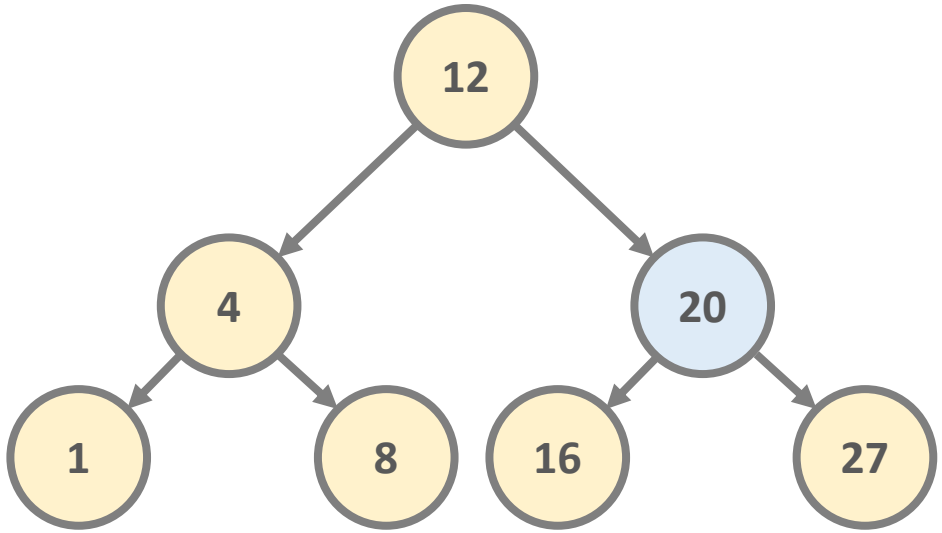
if node right child is not NULL:
return max(node right child)

return node value



STACK

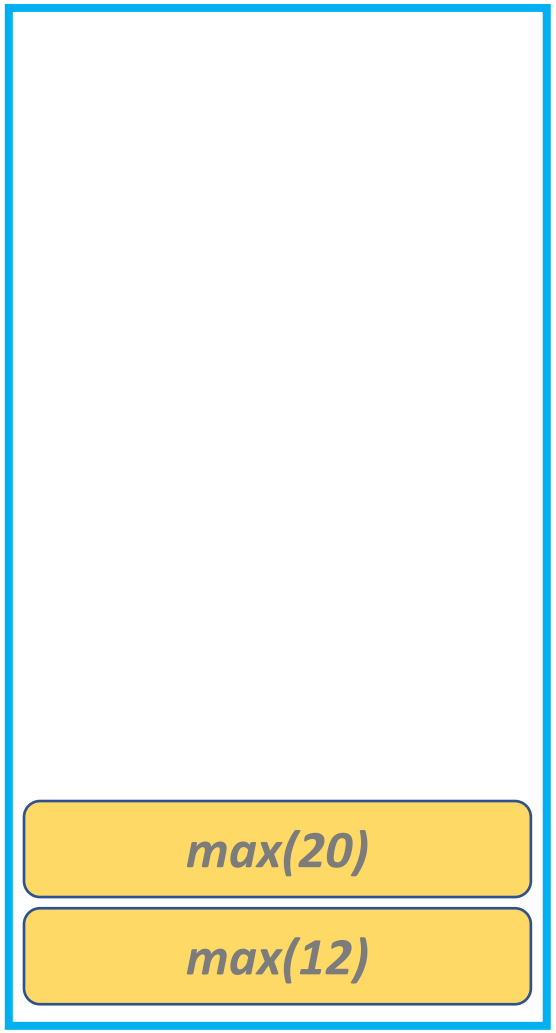
Binary Search Trees



max(node):

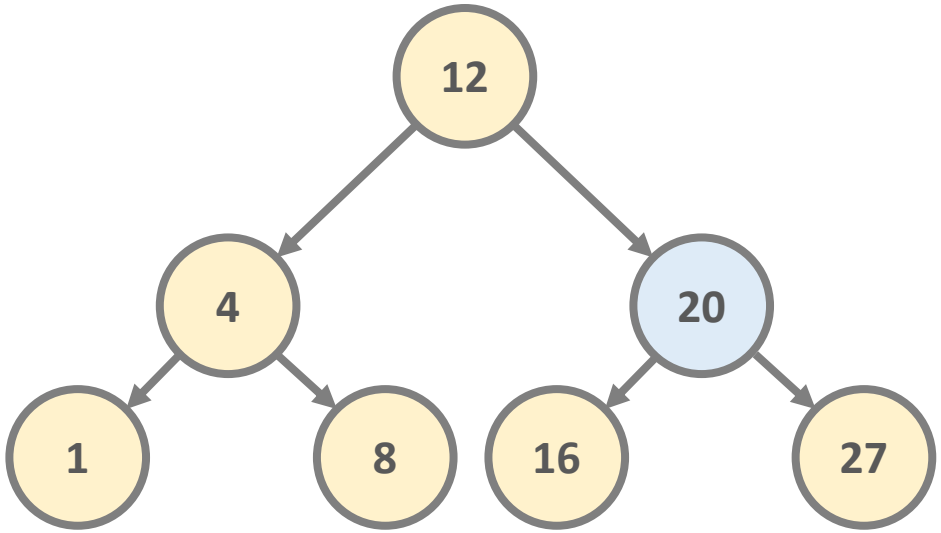
if node right child is not NULL:
return max(node right child)

return node value



STACK

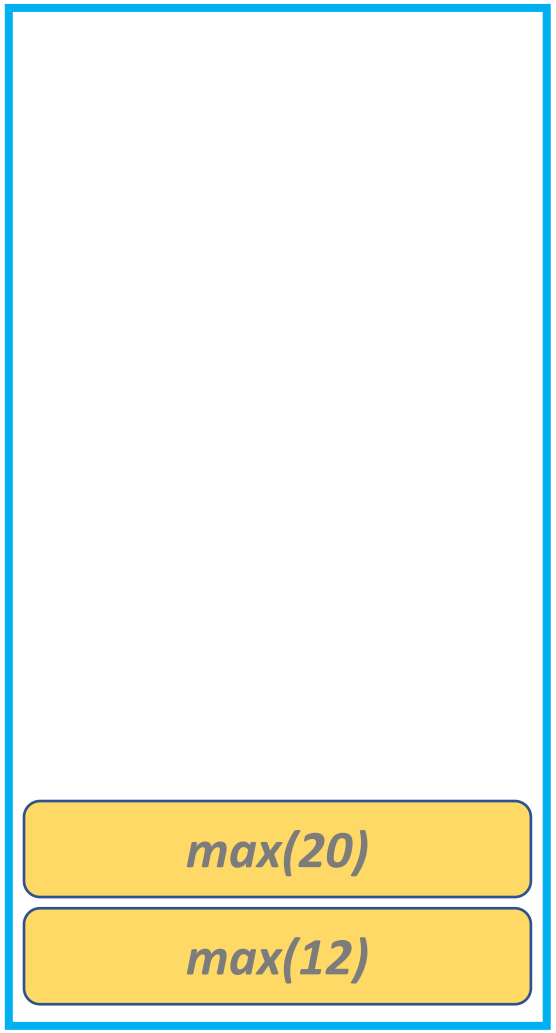
Binary Search Trees



max(node):

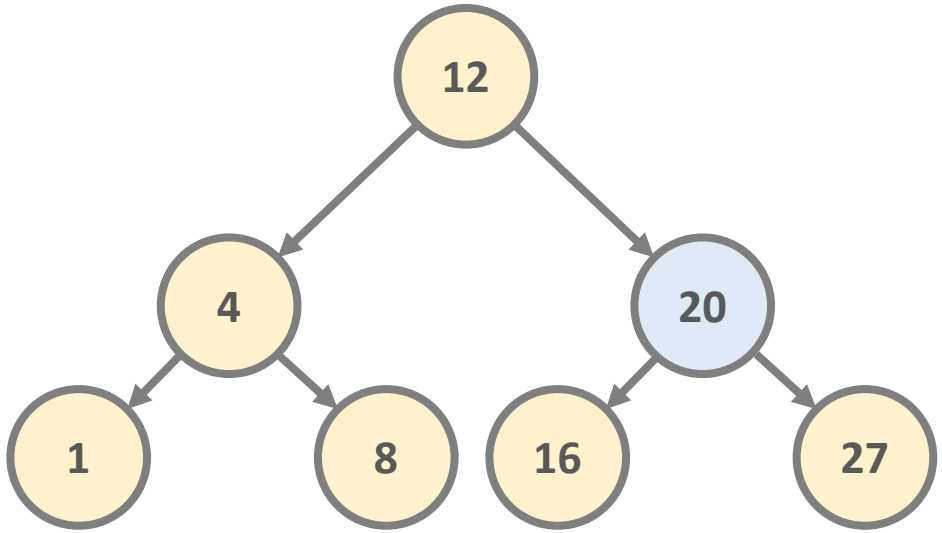
if node right child is not NULL:
return max(node right child)

return node value



STACK

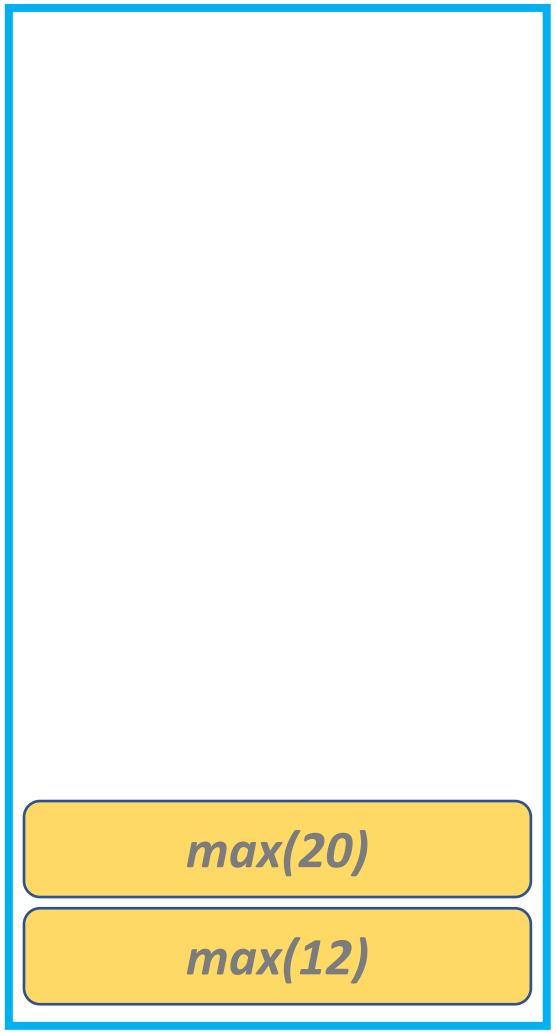
Binary Search Trees



max(node):

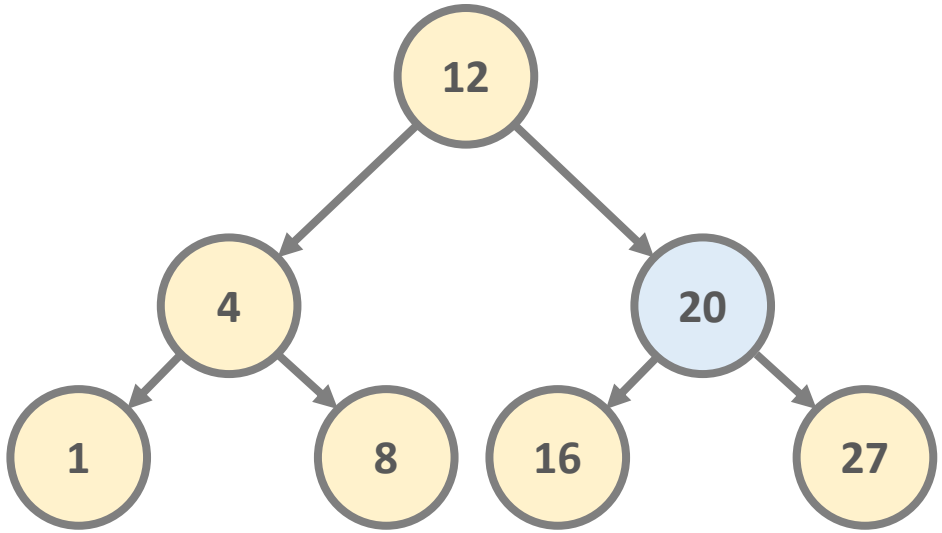
if node right child is not NULL:
return max(node right child)

return node value



STACK

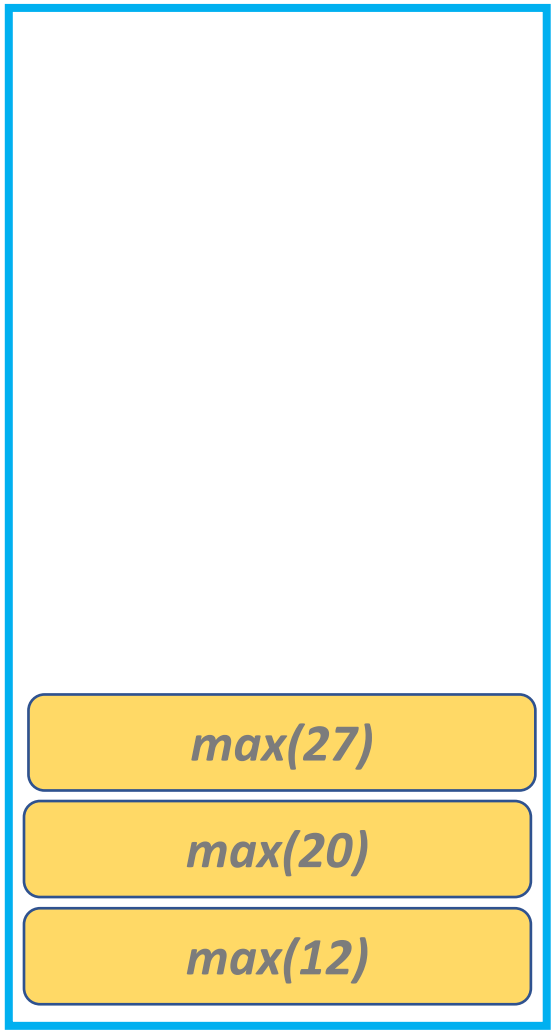
Binary Search Trees



max(node):

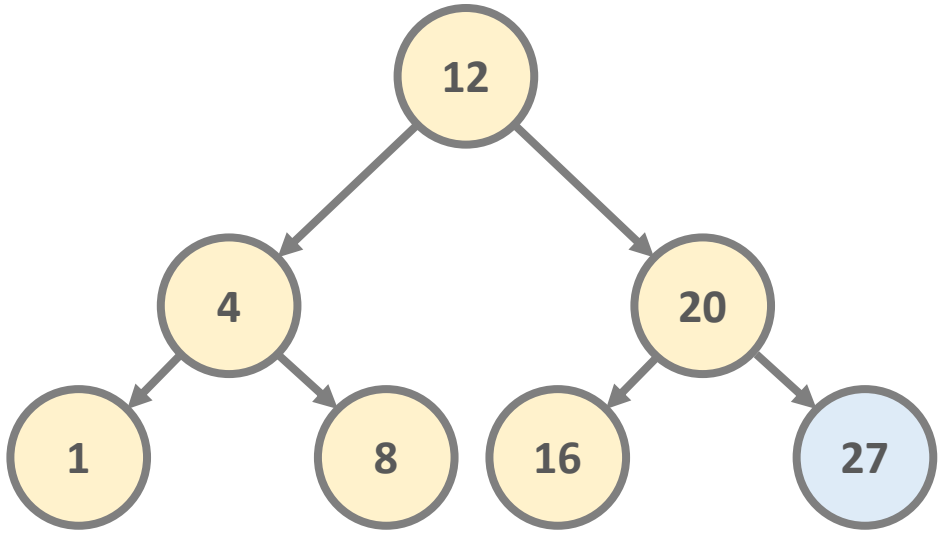
if node right child is not NULL:
return max(node right child)

return node value



STACK

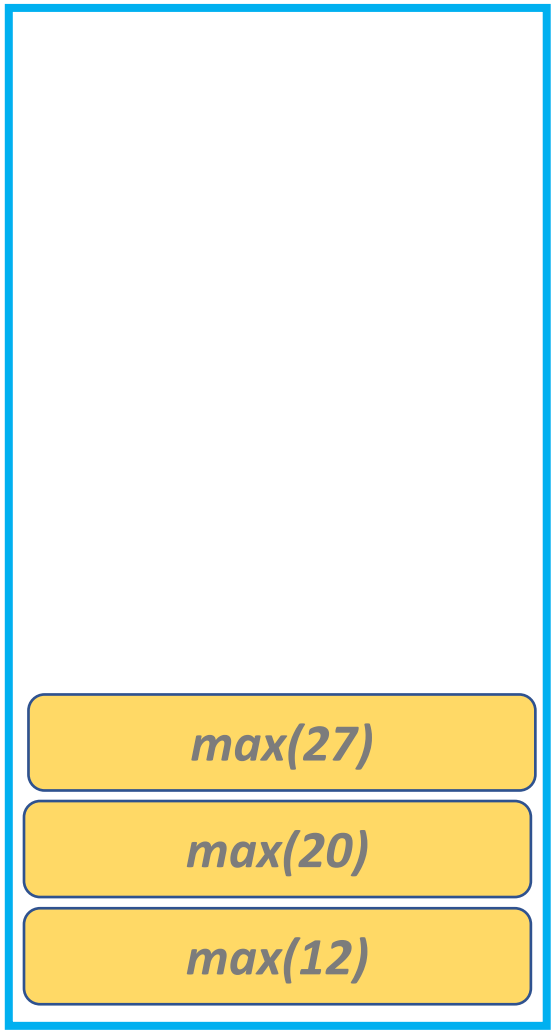
Binary Search Trees



max(node):

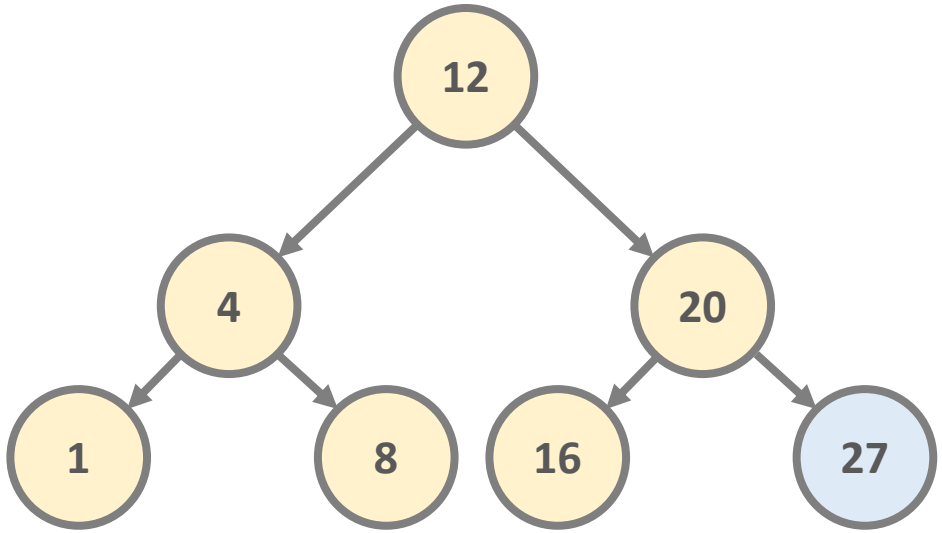
if node right child is not NULL:
return max(node right child)

return node value



STACK

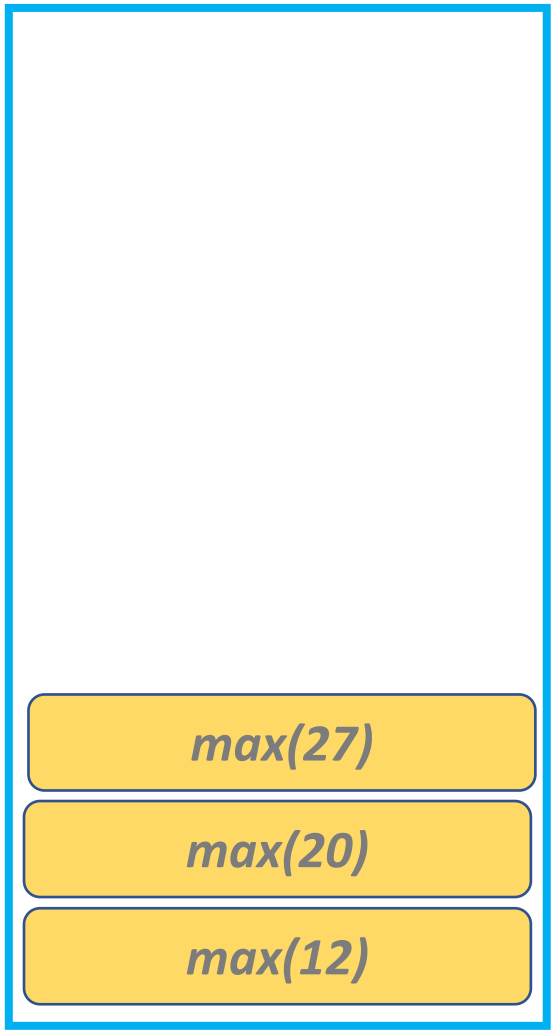
Binary Search Trees



max(node):

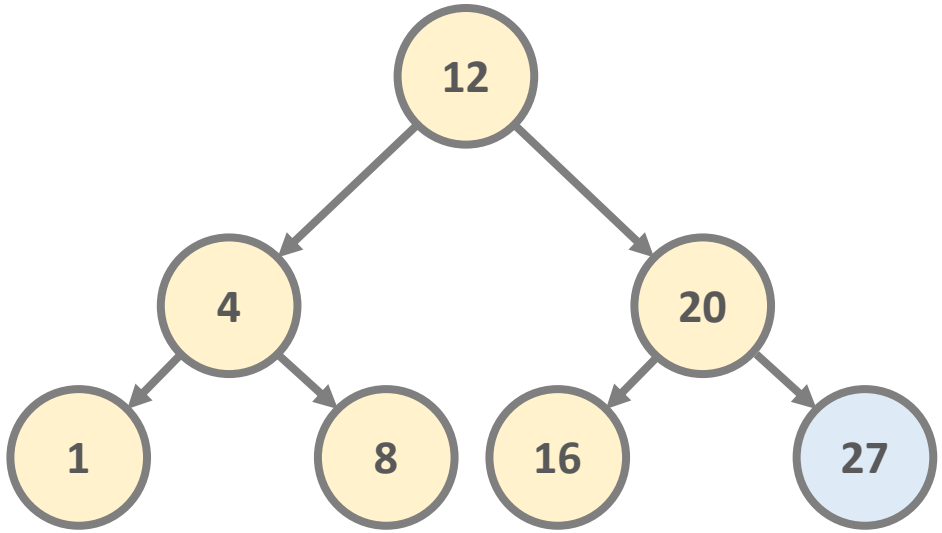
if node right child is not NULL:
return max(node right child)

return node value



STACK

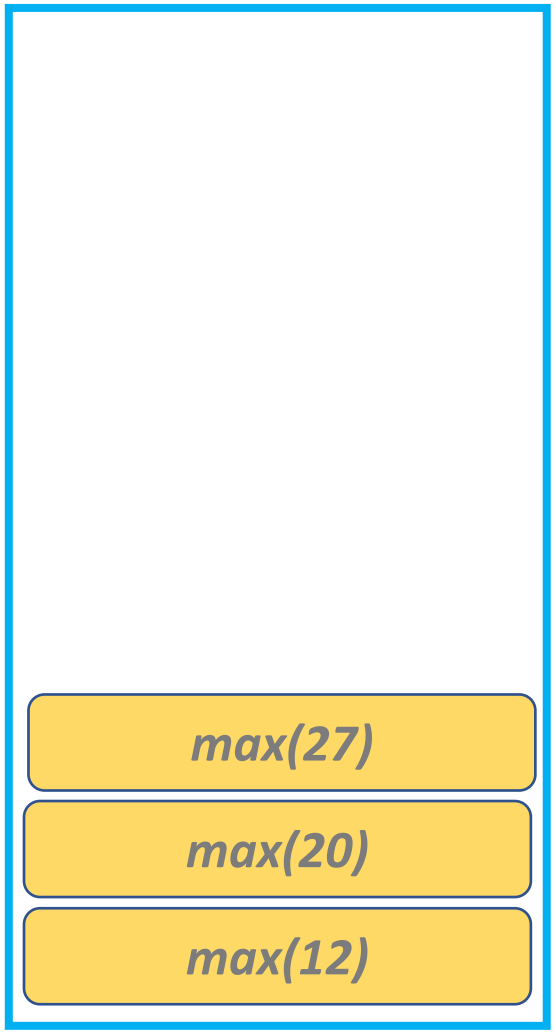
Binary Search Trees



max(node):

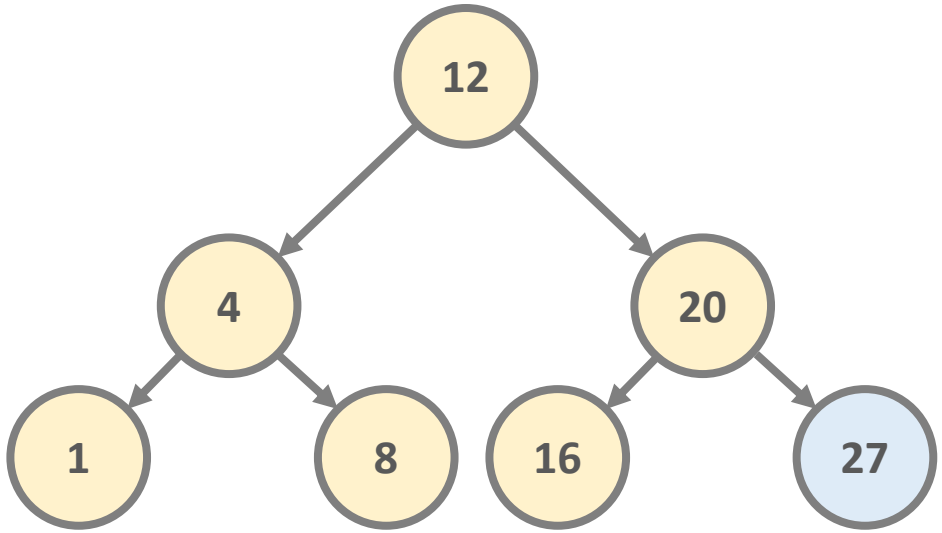
if node right child is not NULL:
return max(node right child)

return node value



STACK

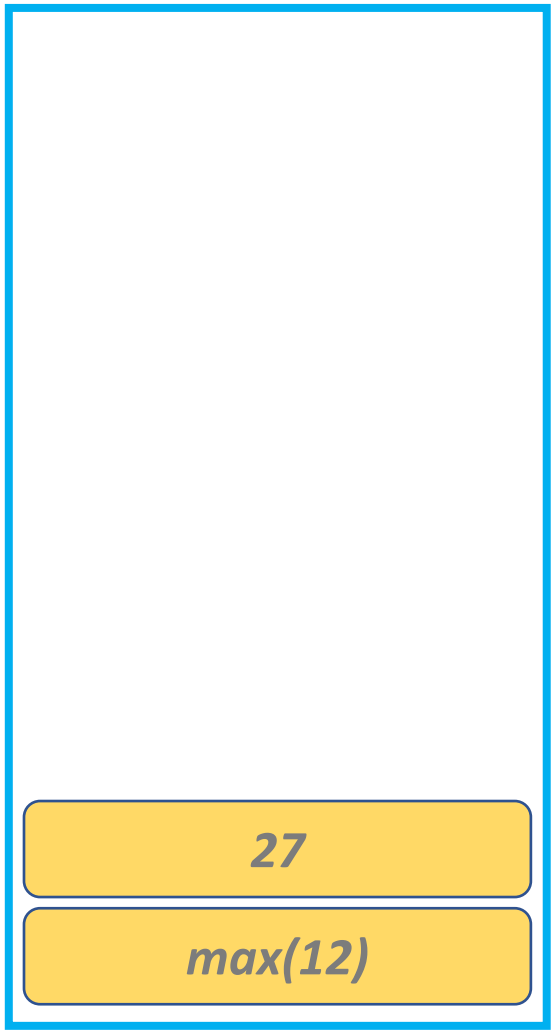
Binary Search Trees



max(node):

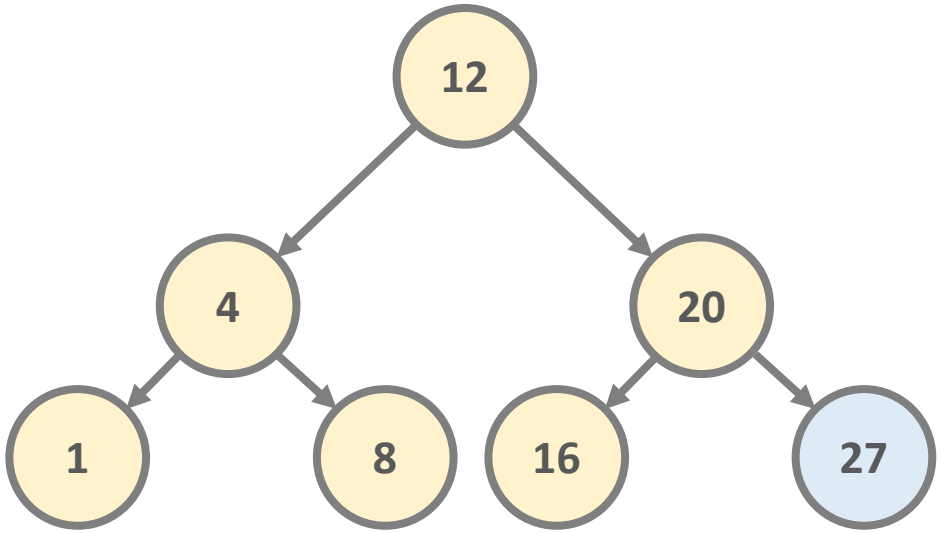
if node right child is not NULL:
return max(node right child)

return node value



STACK

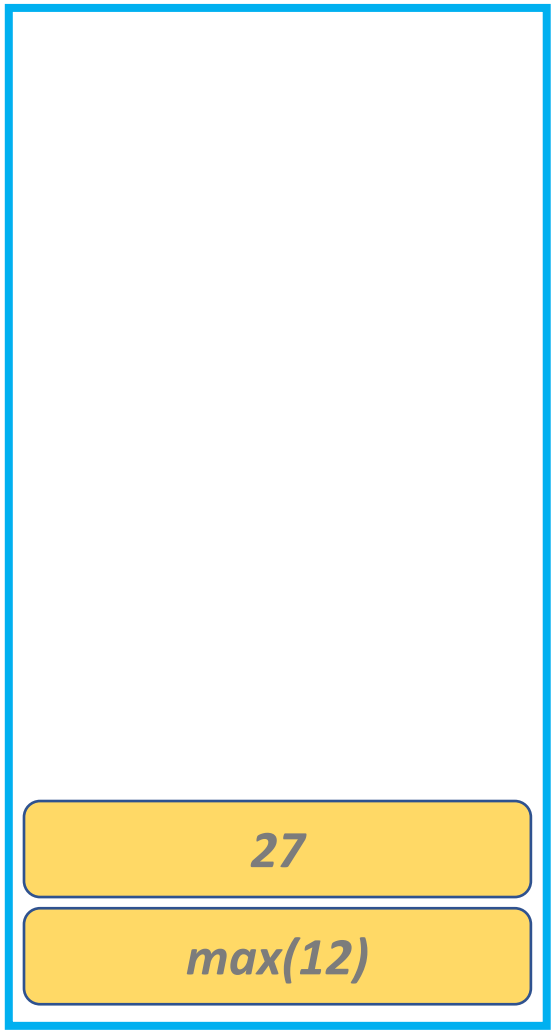
Binary Search Trees



max(node):

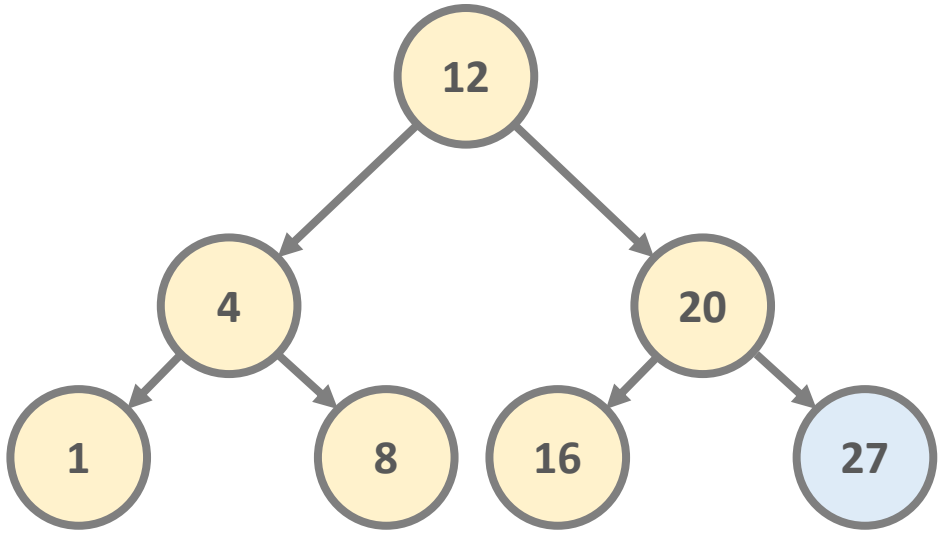
if node right child is not NULL:
return max(node right child)

return node value



STACK

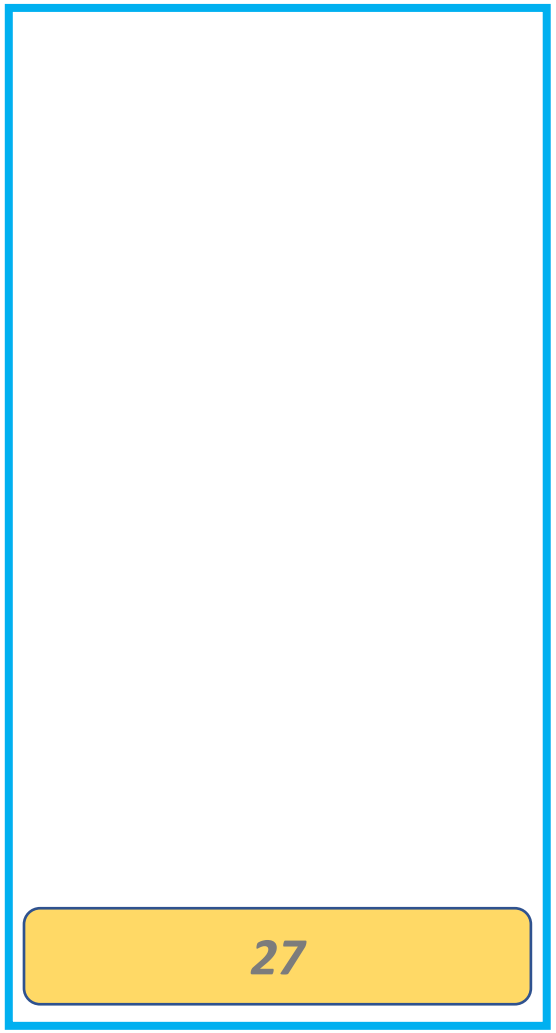
Binary Search Trees



max(node):

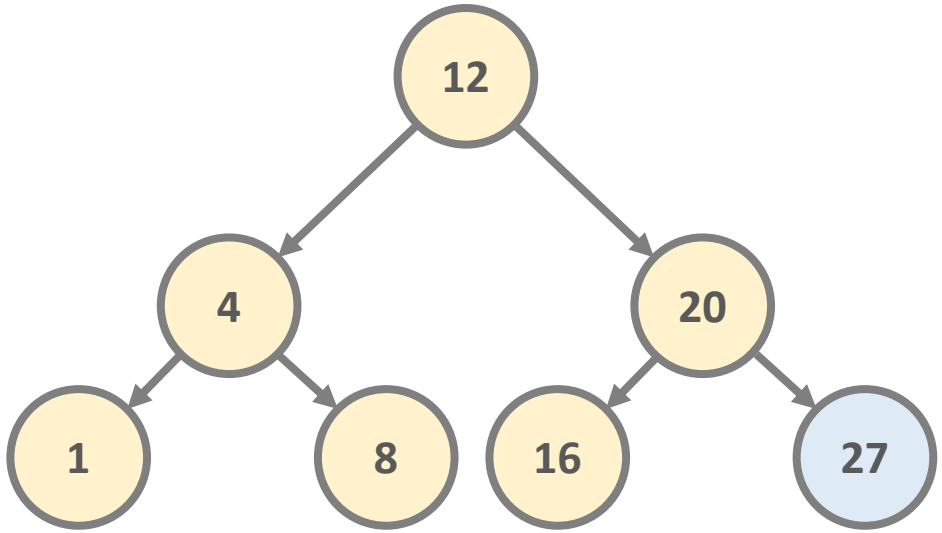
if node right child is not NULL:
return max(node right child)

return node value



STACK

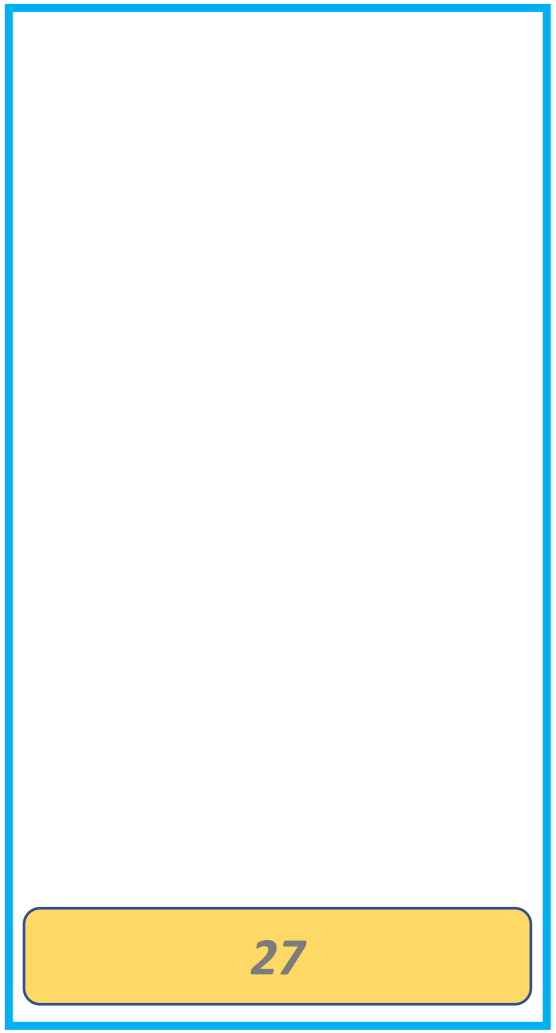
Binary Search Trees



max(node):

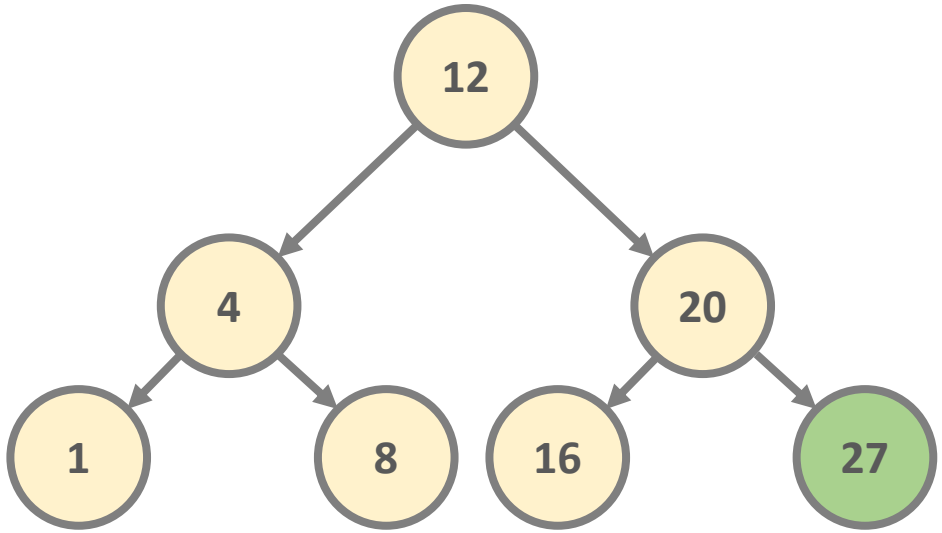
if node right child is not NULL:
return max(node right child)

return node value



STACK

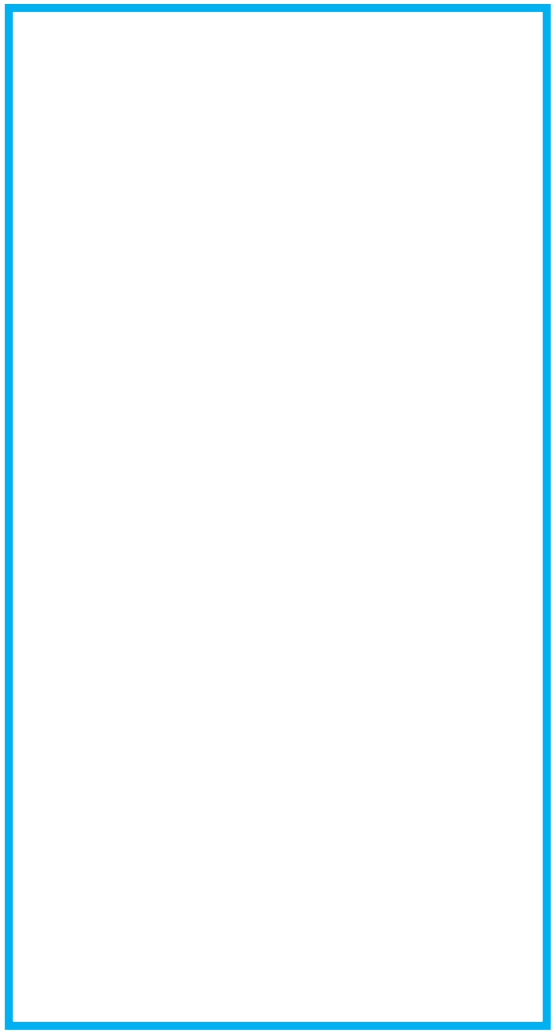
Binary Search Trees



max(node):

if node right child is not NULL:
return max(node right child)

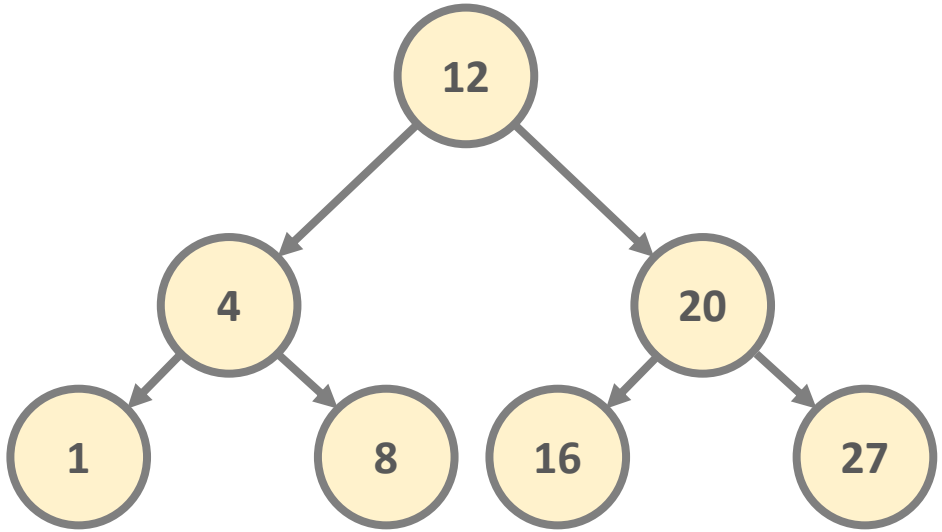
return node value



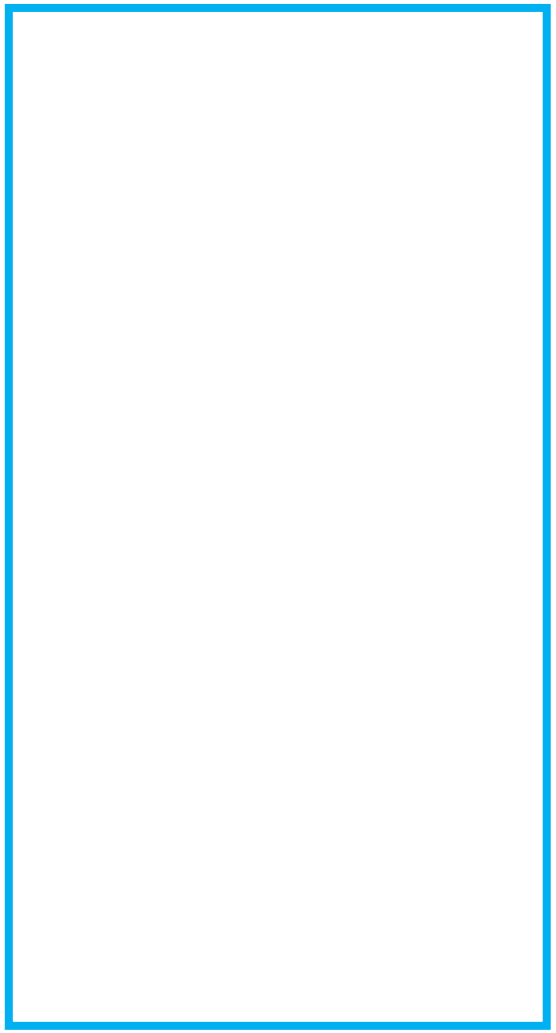
STACK

Stack Memory Visualization
(Algorithmic Problems)

Binary Search Trees

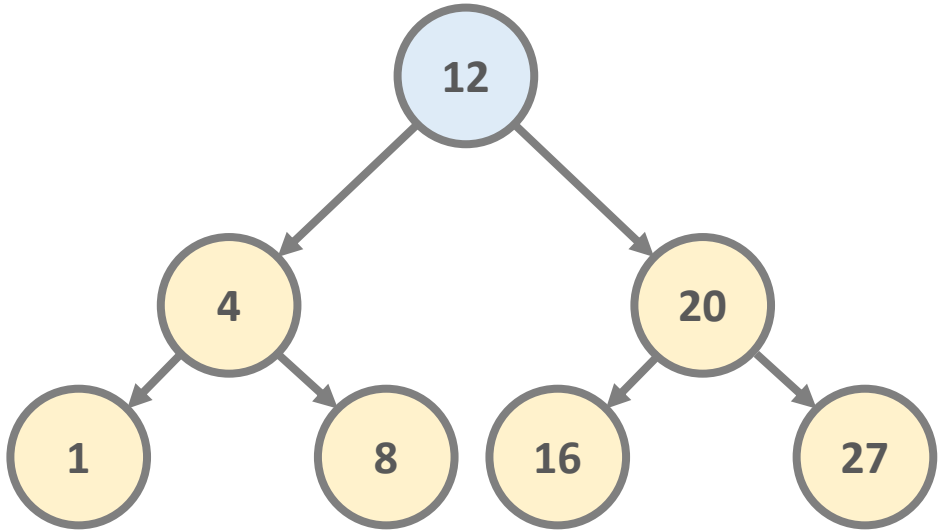



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

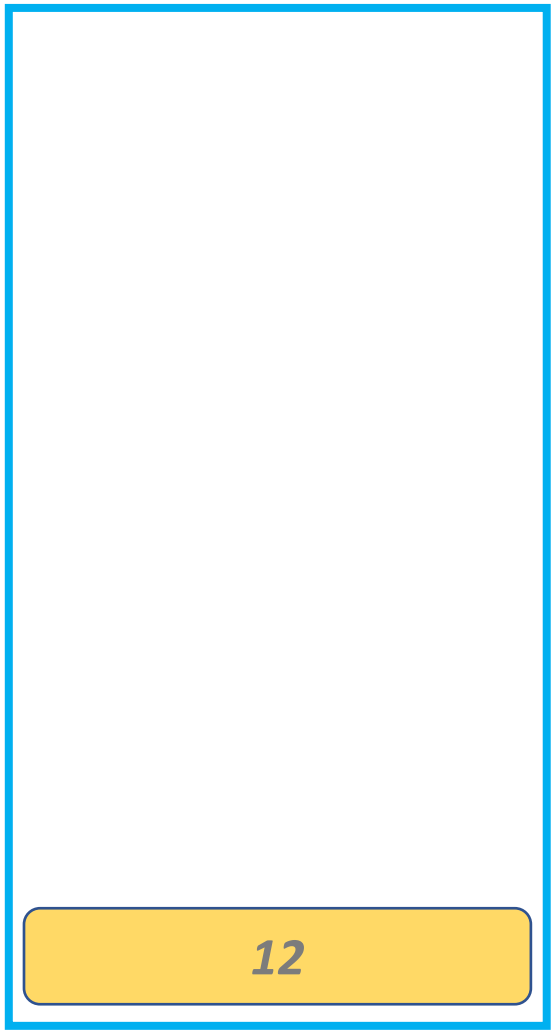


STACK

Binary Search Trees

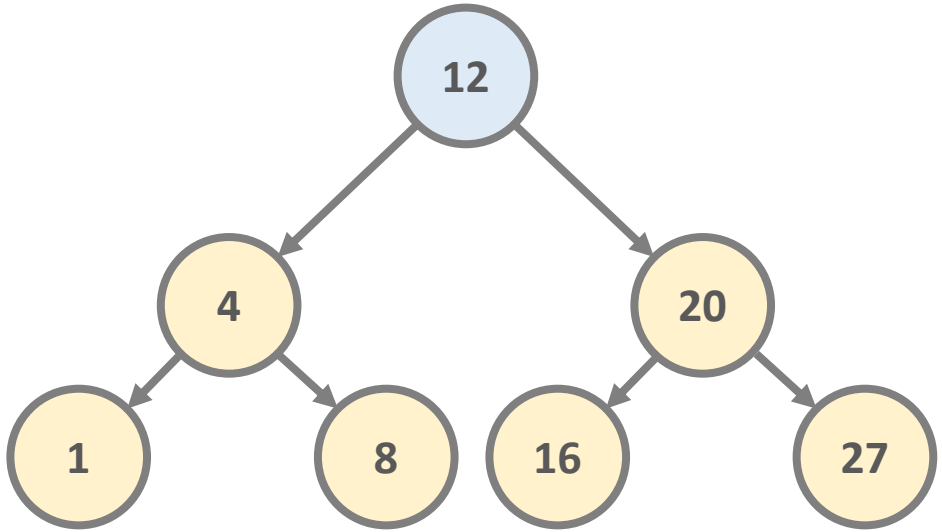



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

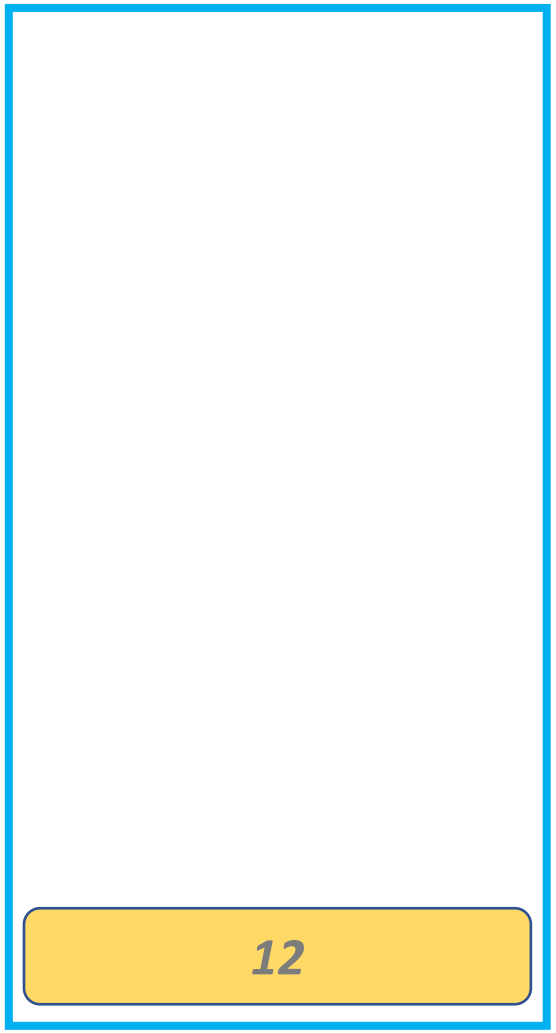


STACK

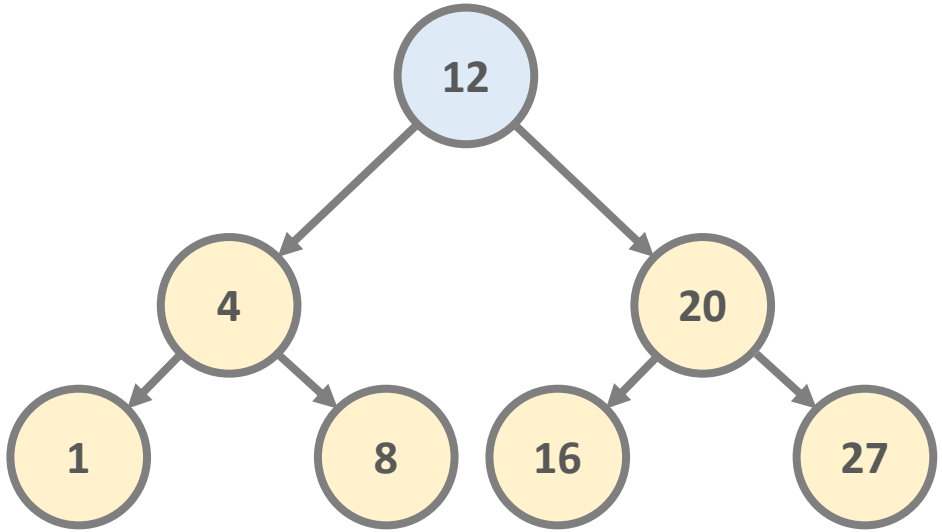
Binary Search Trees



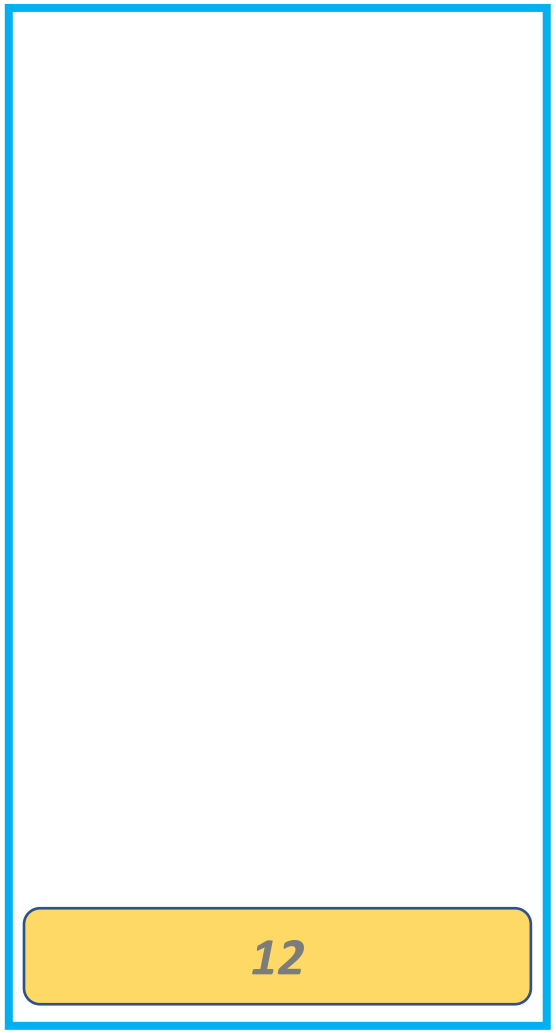

```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



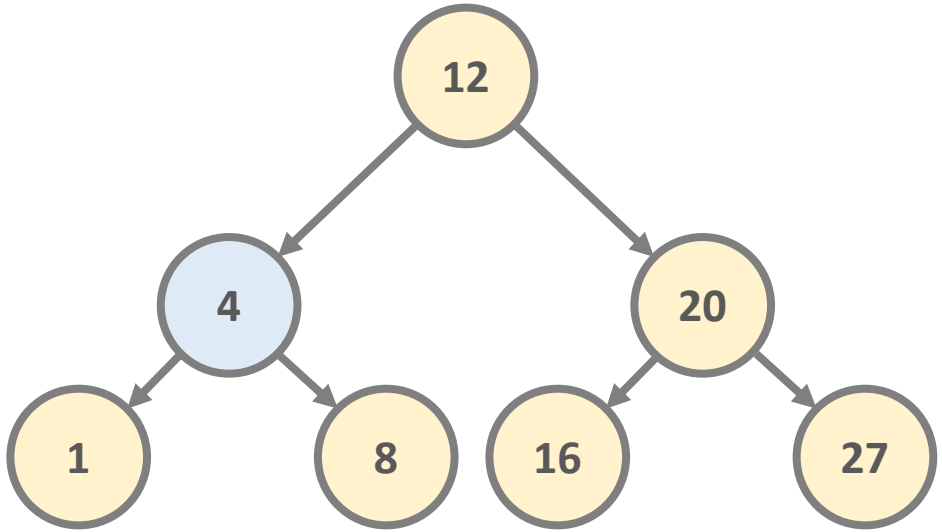
Binary Search Trees



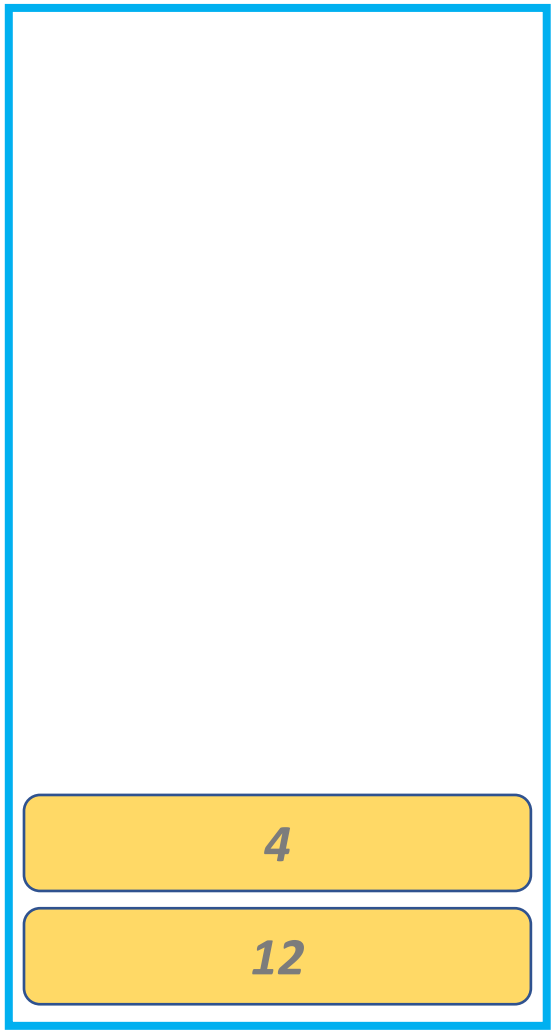

```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



Binary Search Trees

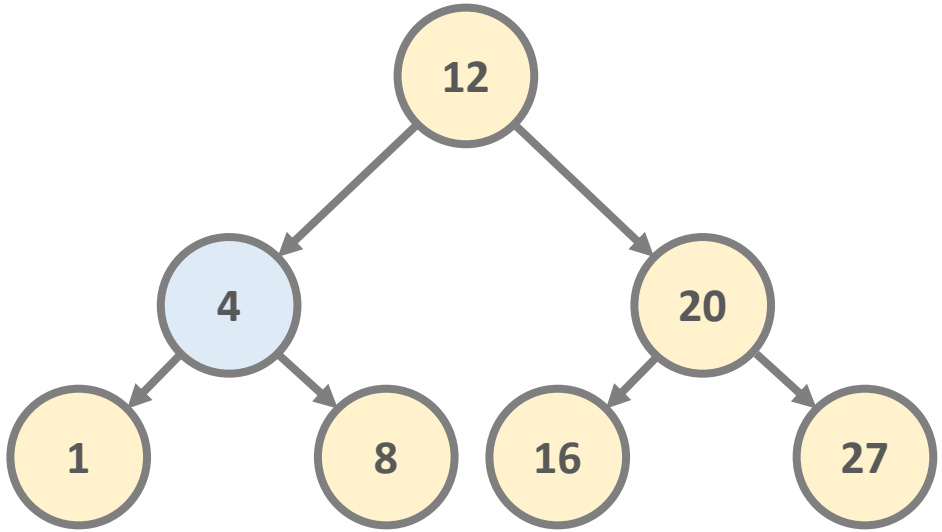



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

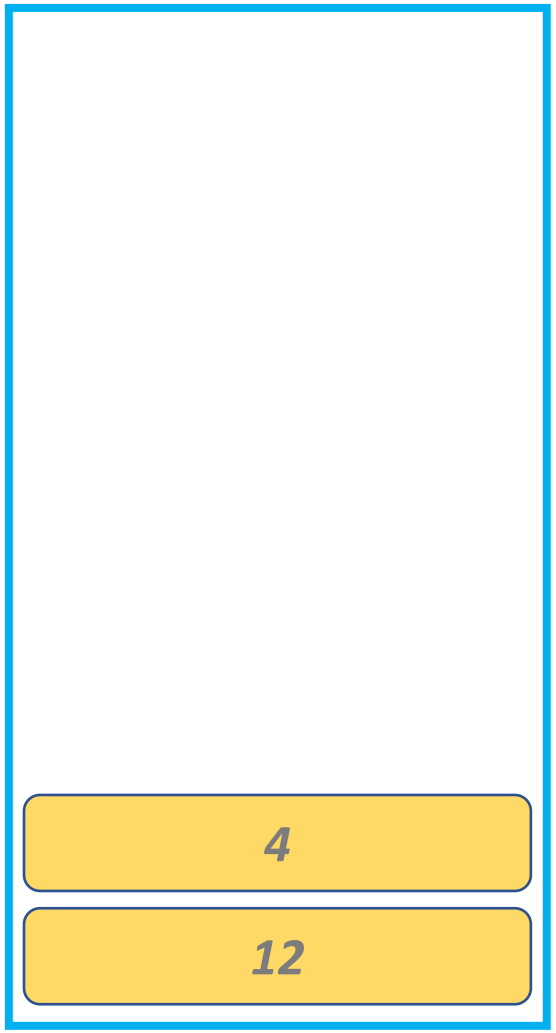


STACK

Binary Search Trees

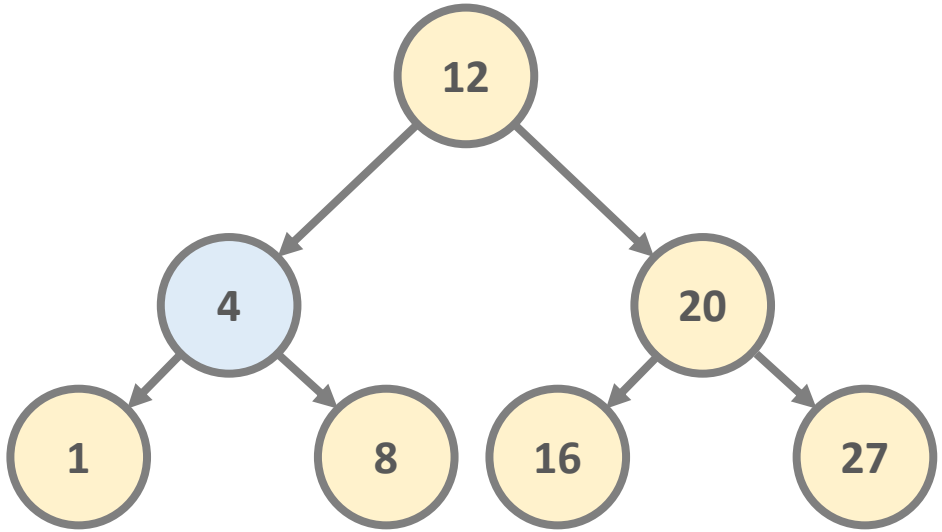



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

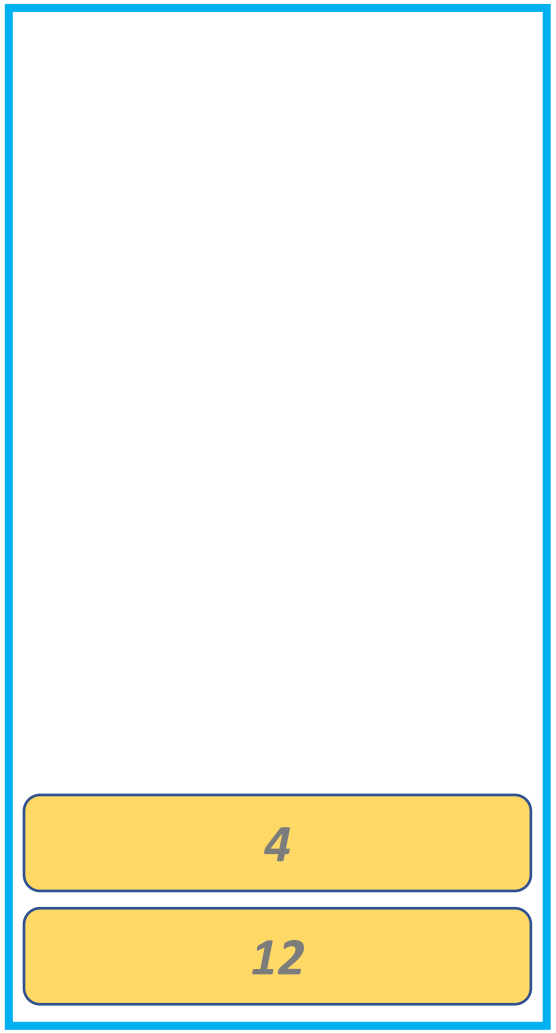


STACK

Binary Search Trees

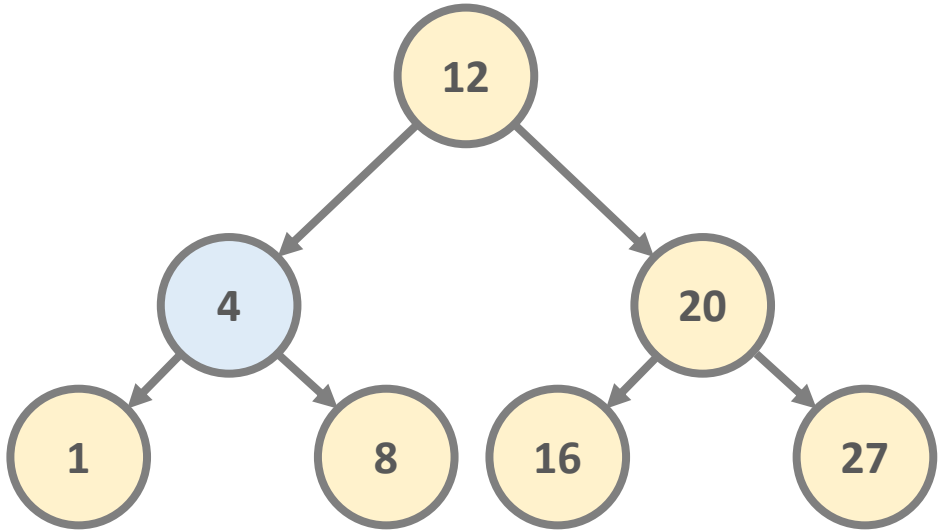



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

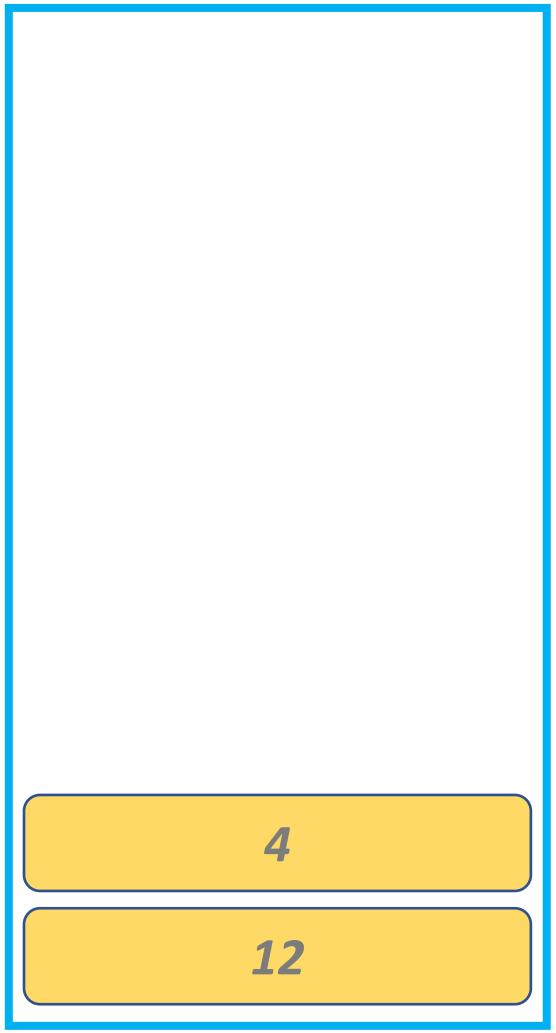


STACK

Binary Search Trees

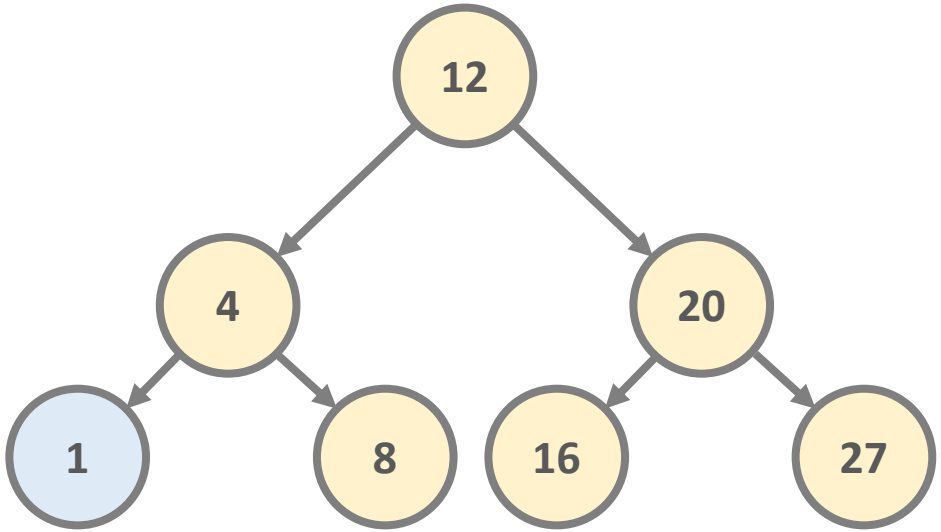



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

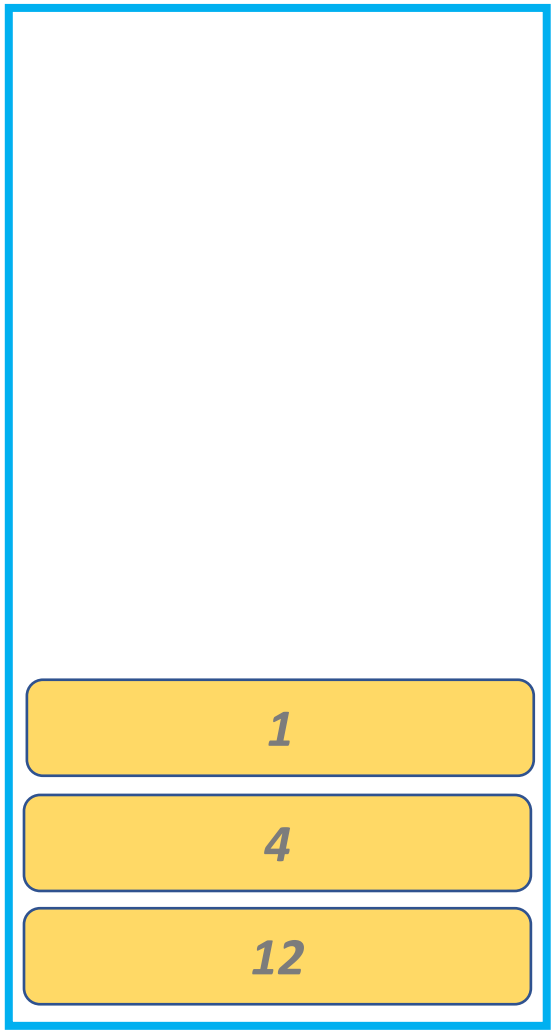


STACK

Binary Search Trees

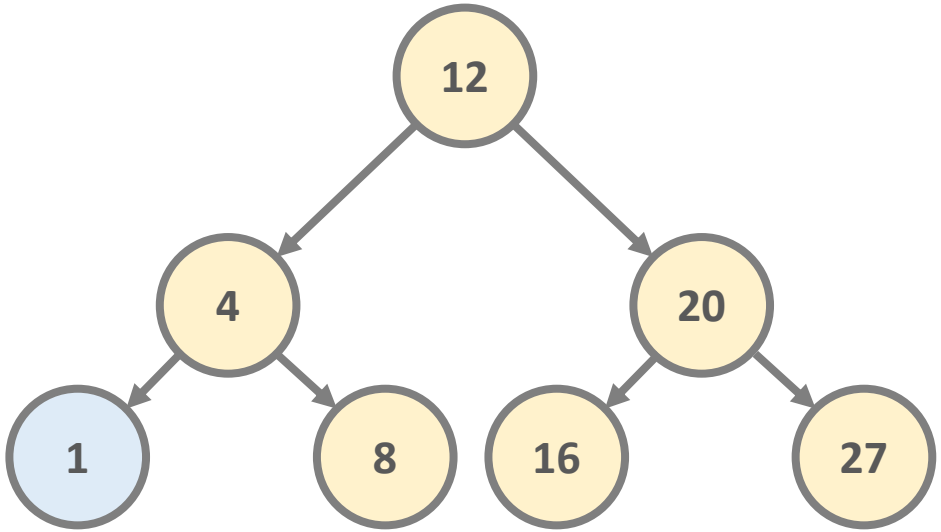



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

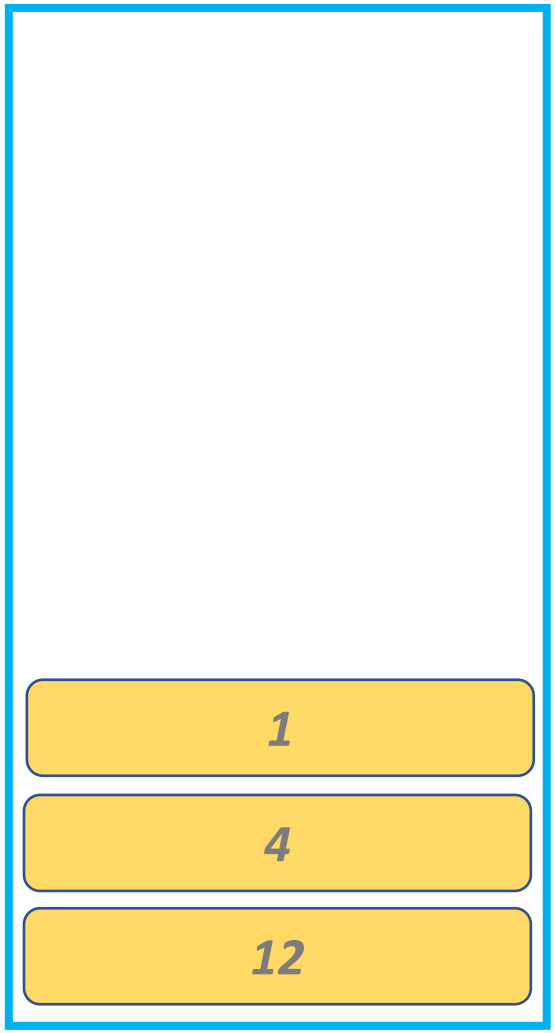


STACK

Binary Search Trees

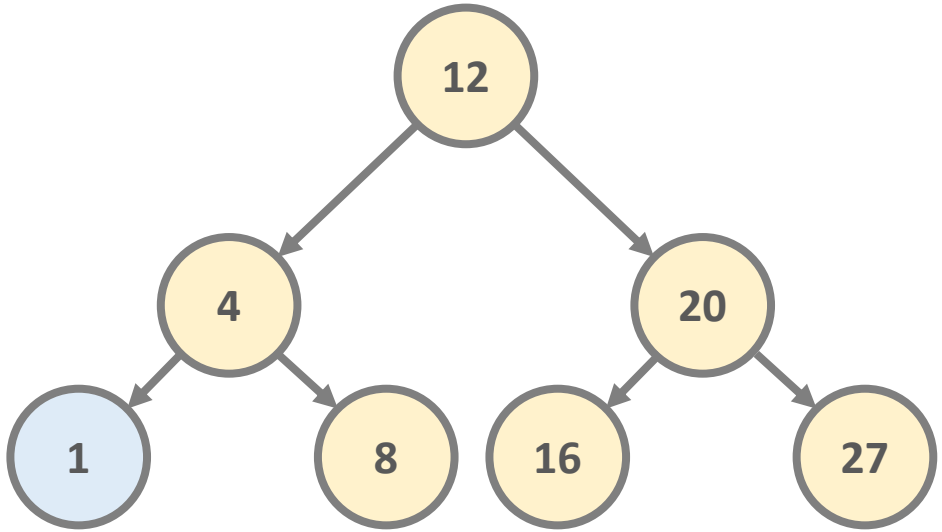



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

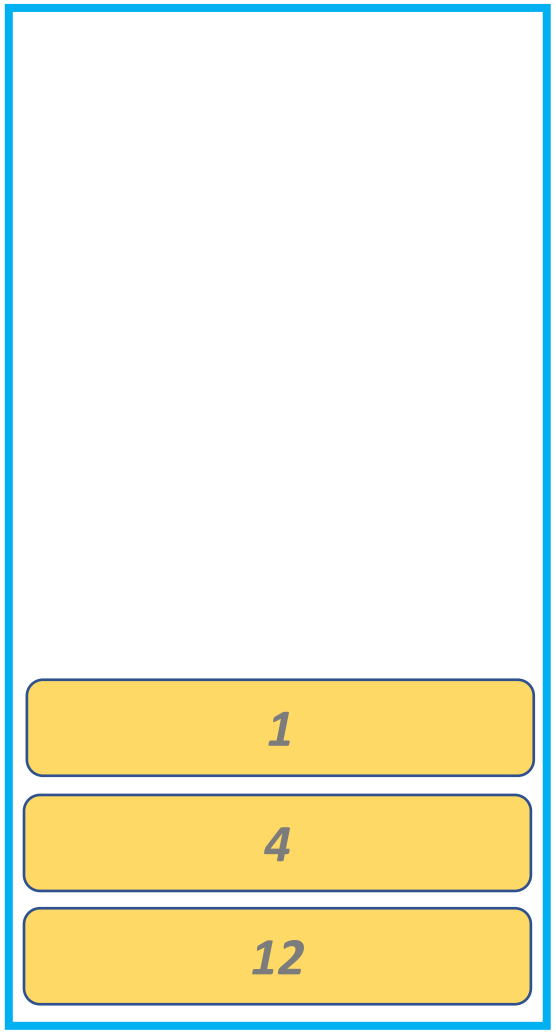


STACK

Binary Search Trees

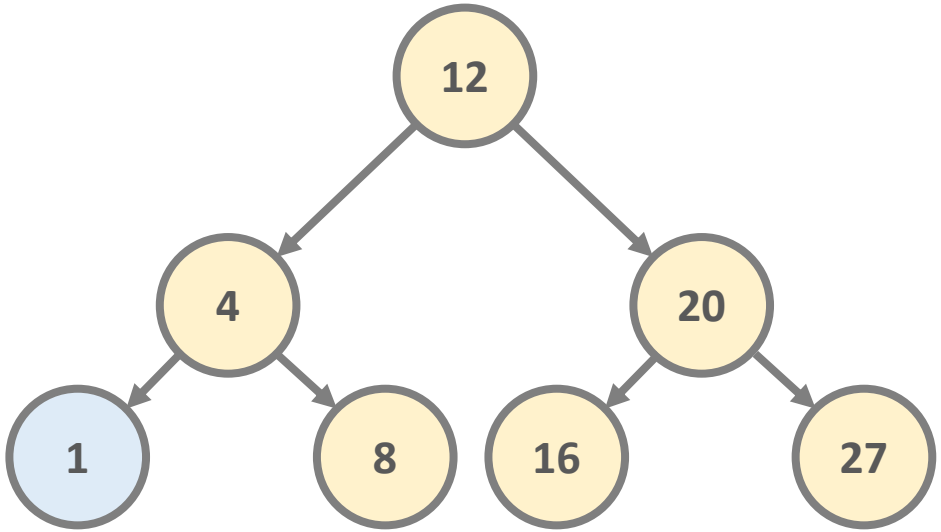



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

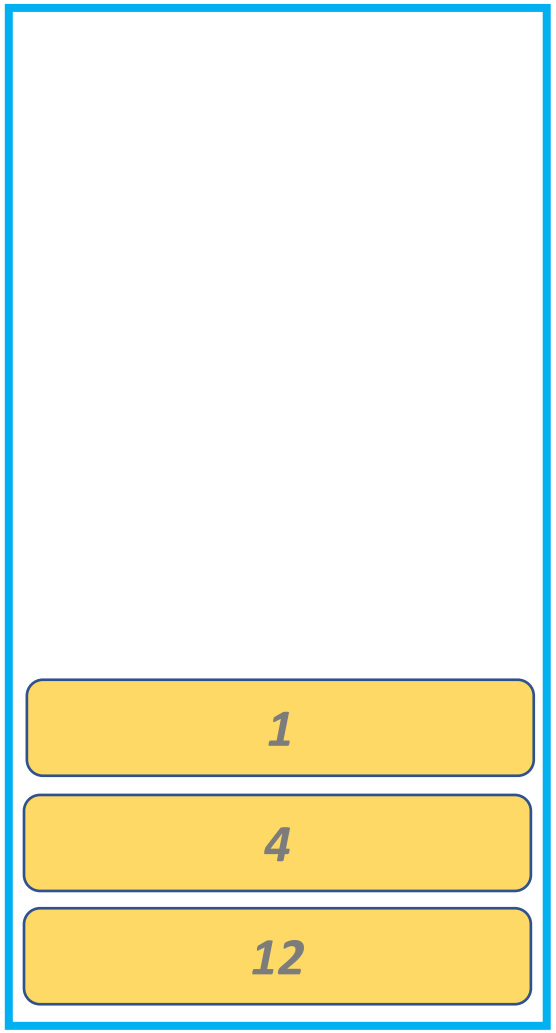


STACK

Binary Search Trees

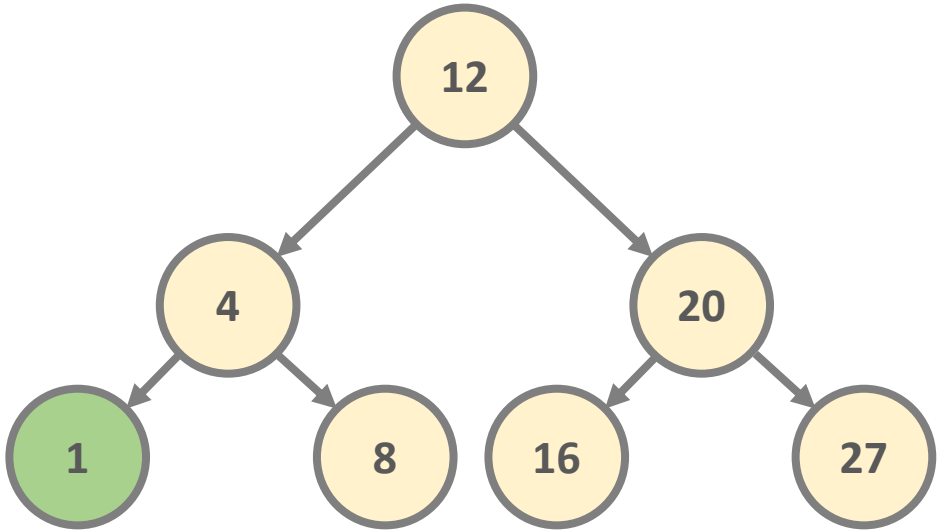



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

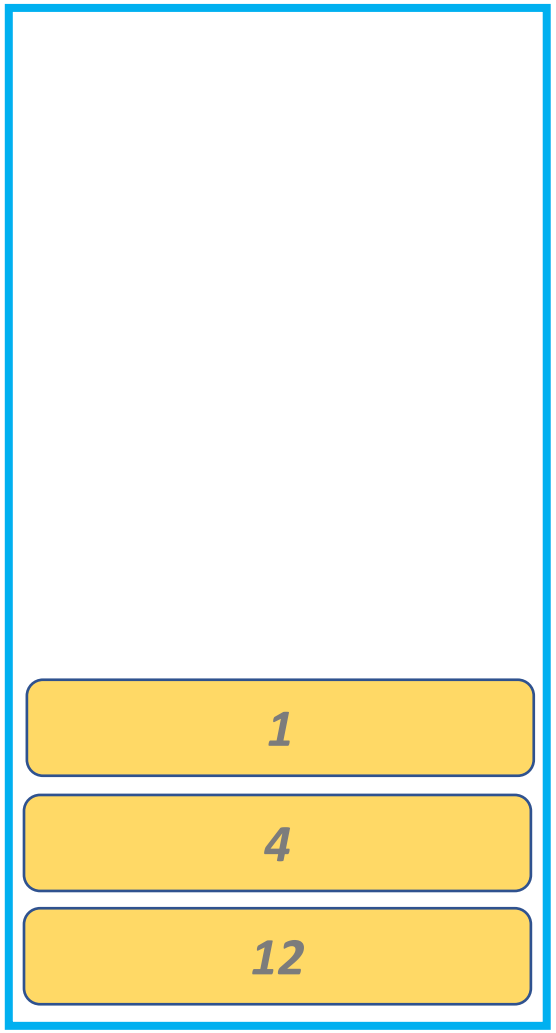


traverse(node):

if node left child is not NULL:
traverse(node left child)

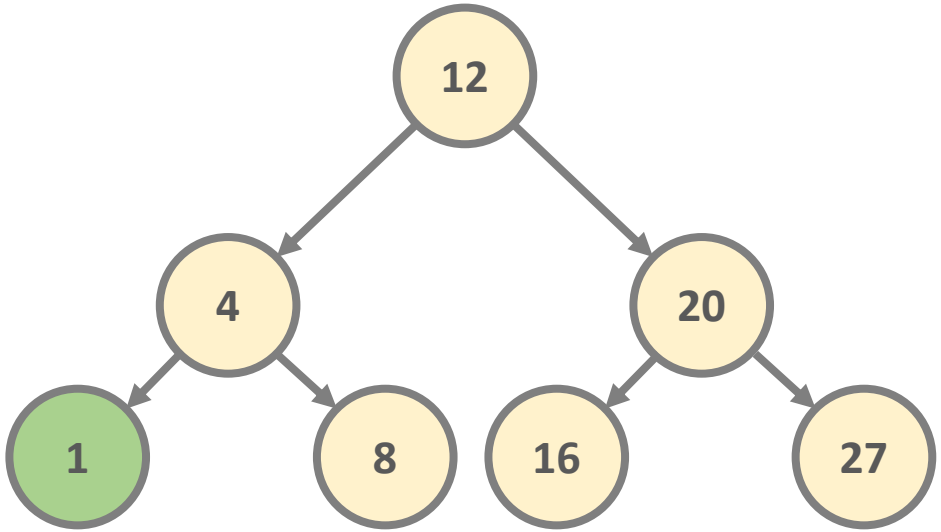
print node.data

if node right child is not NULL:
traverse(node right child)

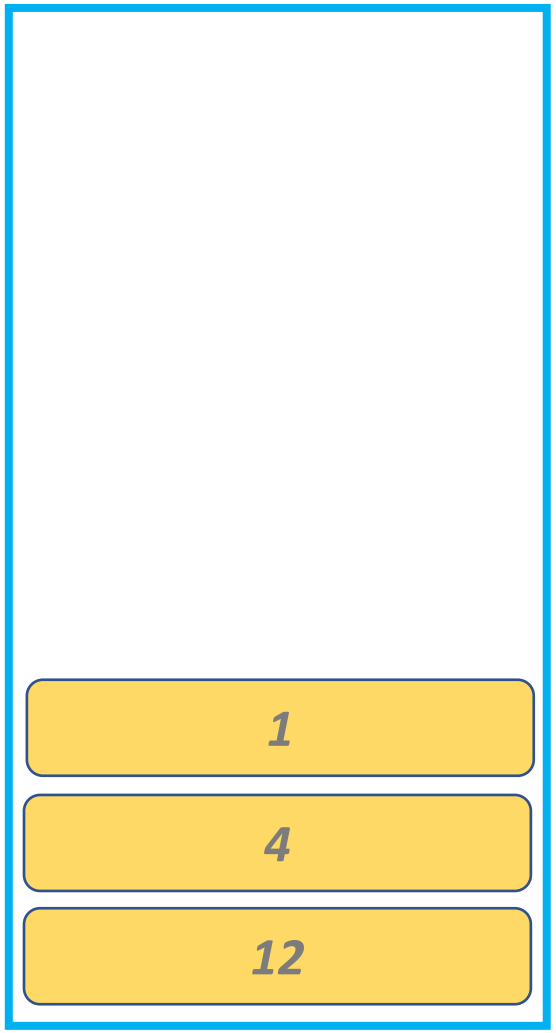


STACK

Binary Search Trees

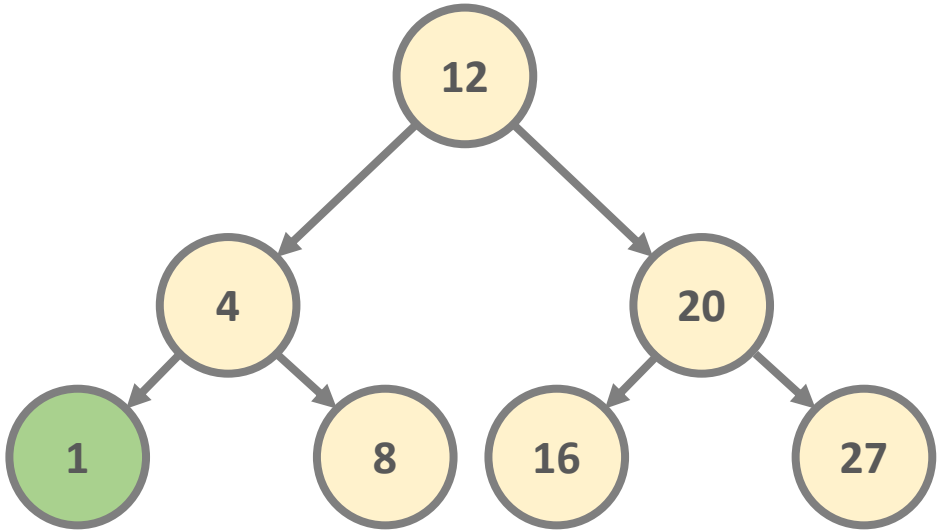



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

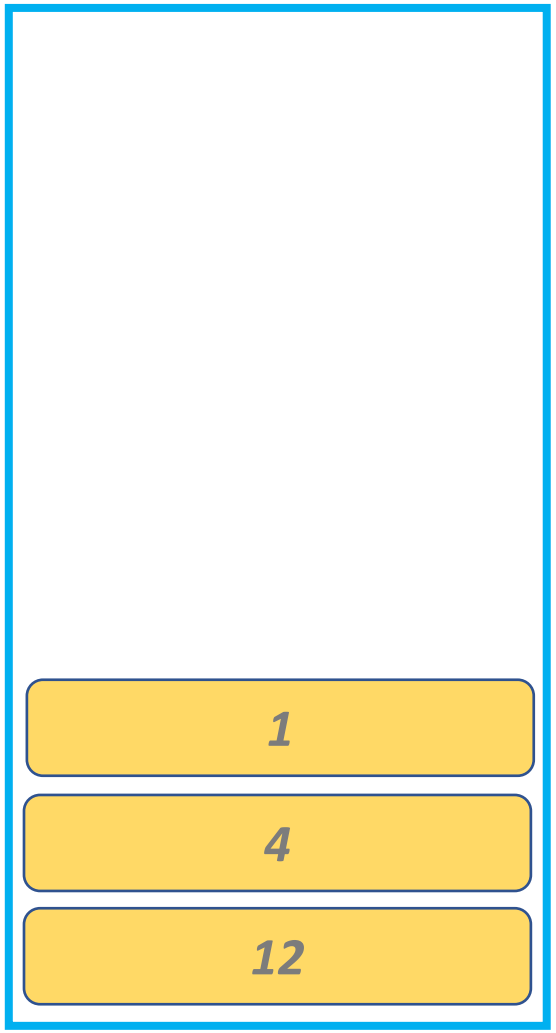


traverse(node):

if node left child is not NULL:
traverse(node left child)

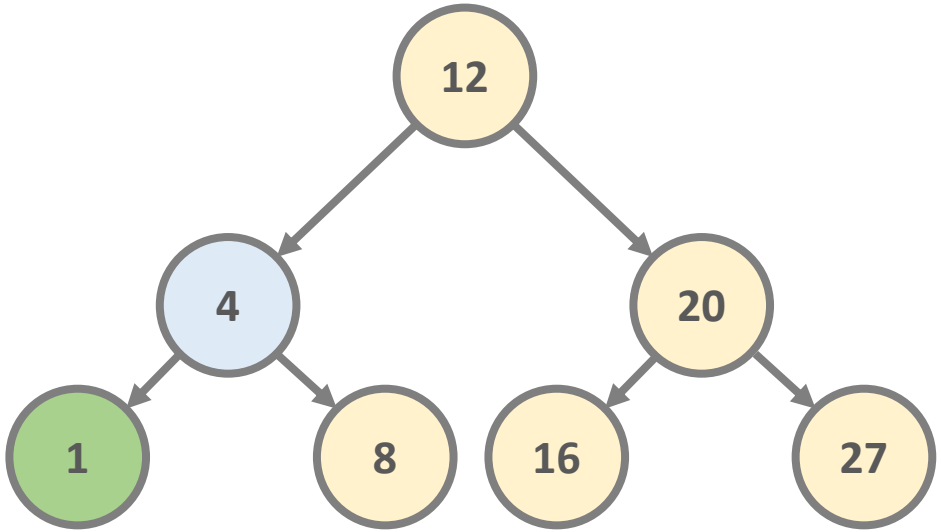
print node.data

if node right child is not NULL:
traverse(node right child)

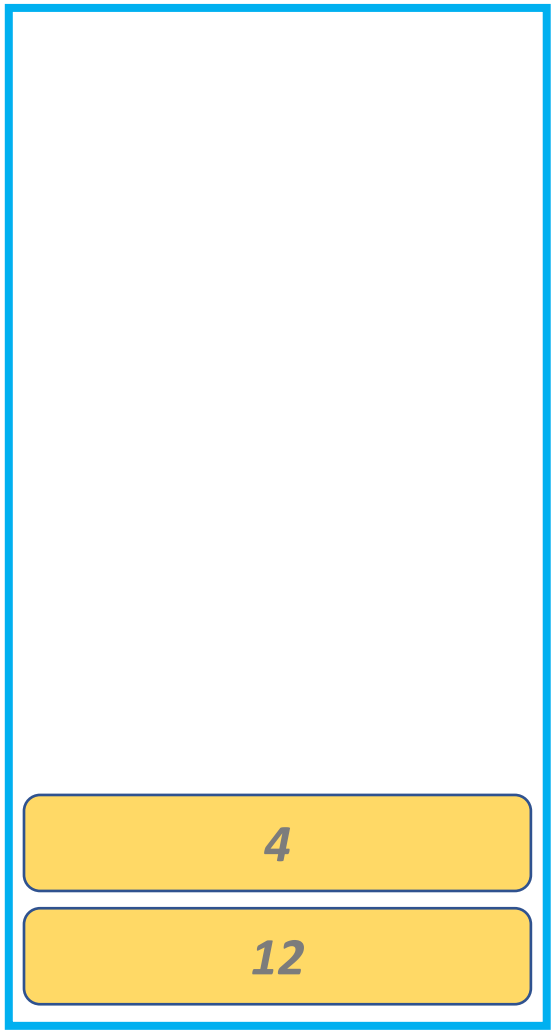


STACK

Binary Search Trees

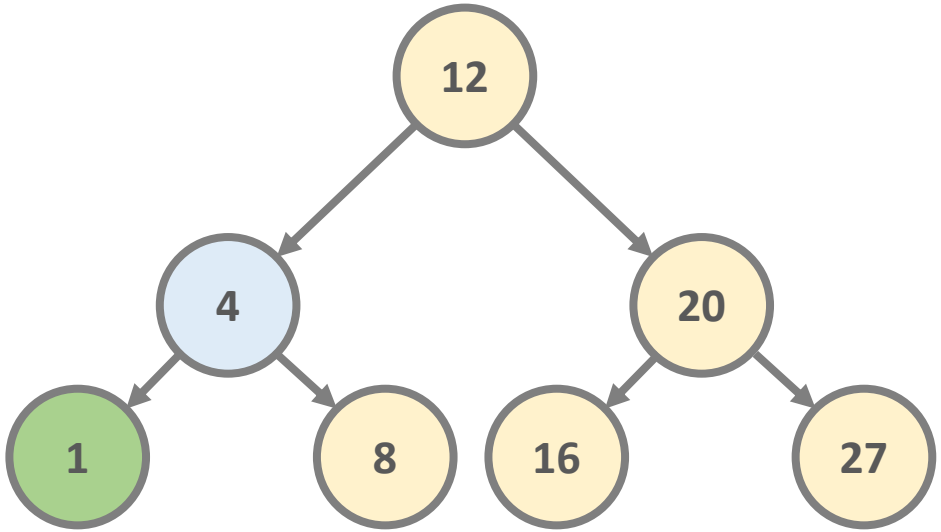



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

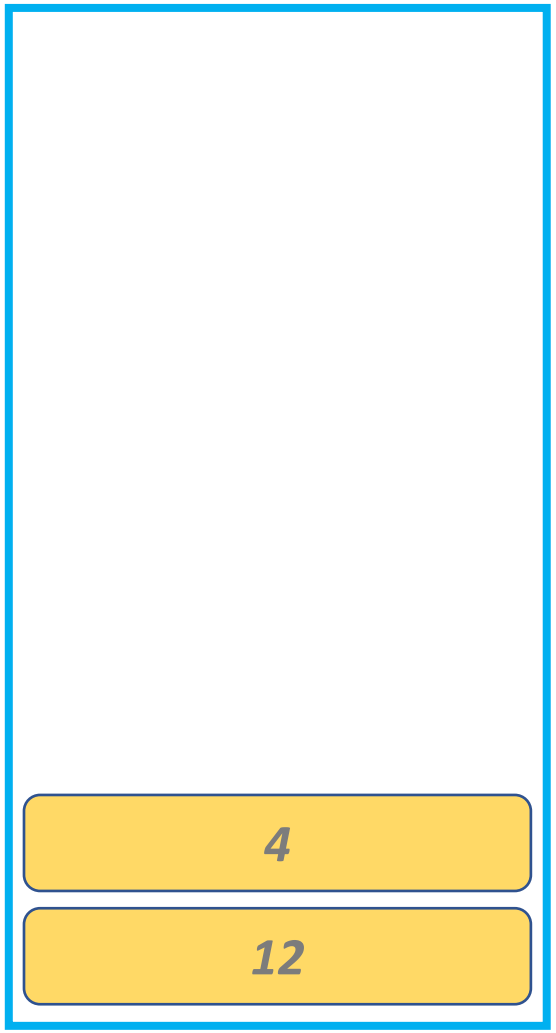


STACK

Binary Search Trees

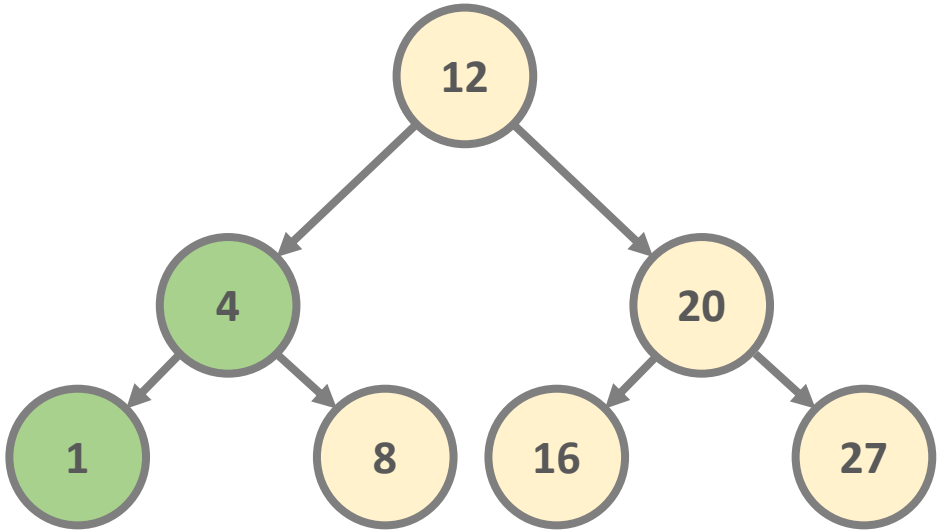



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

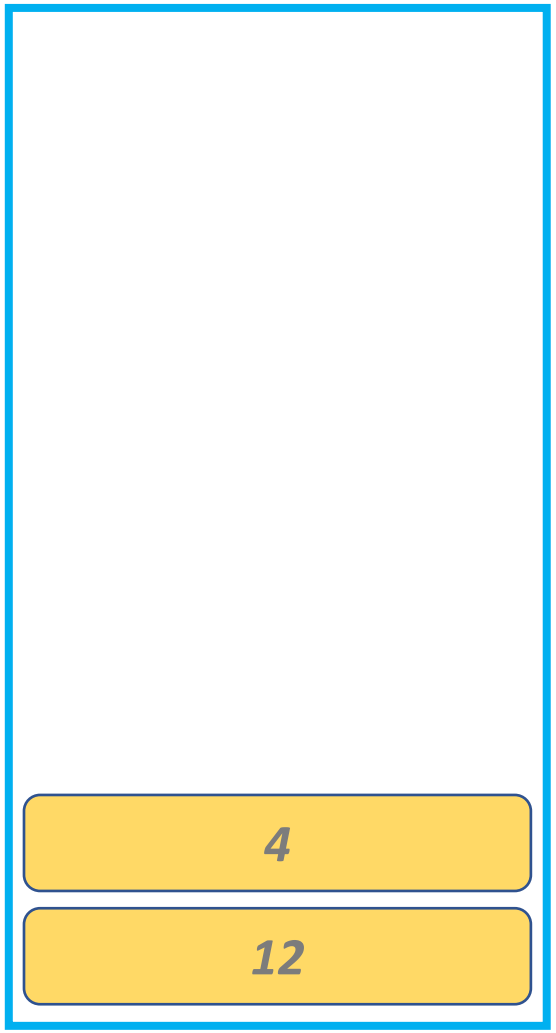


STACK

Binary Search Trees

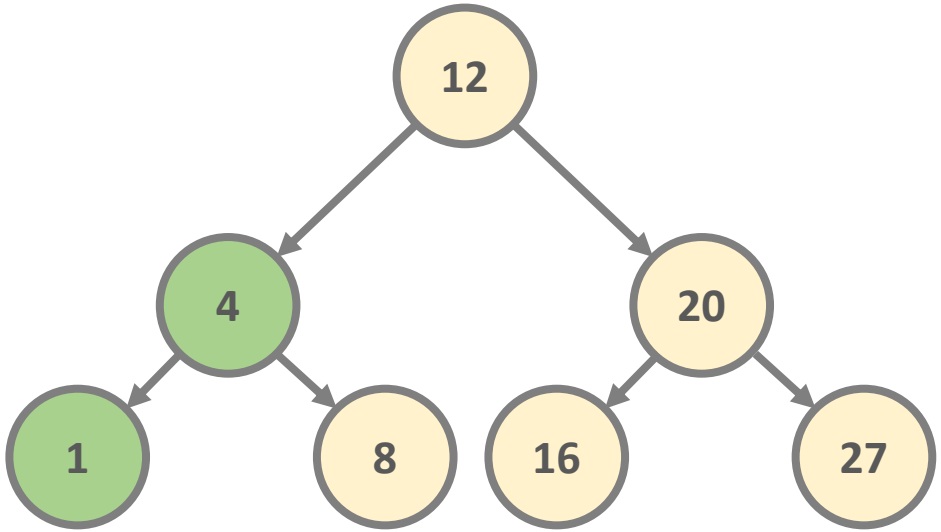



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

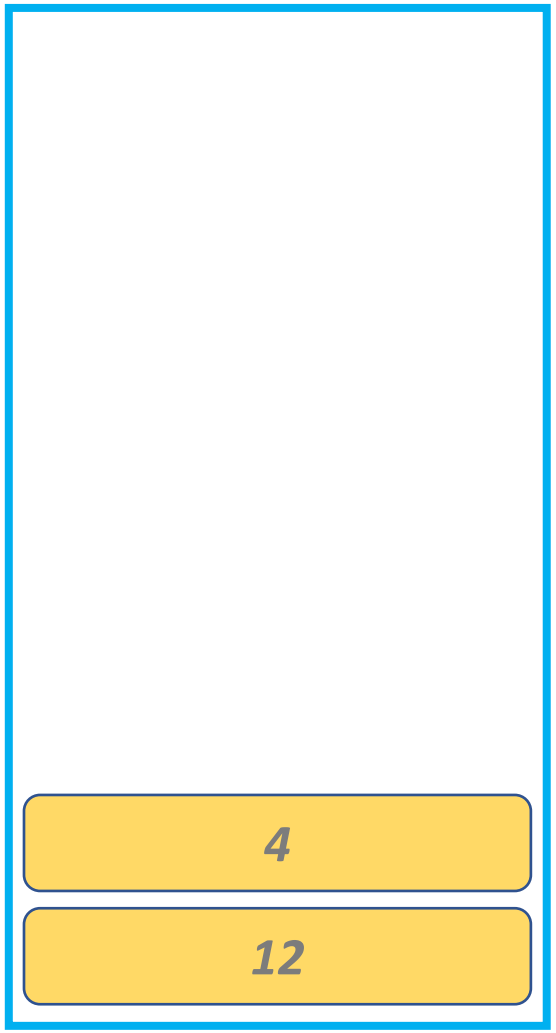


STACK

Binary Search Trees

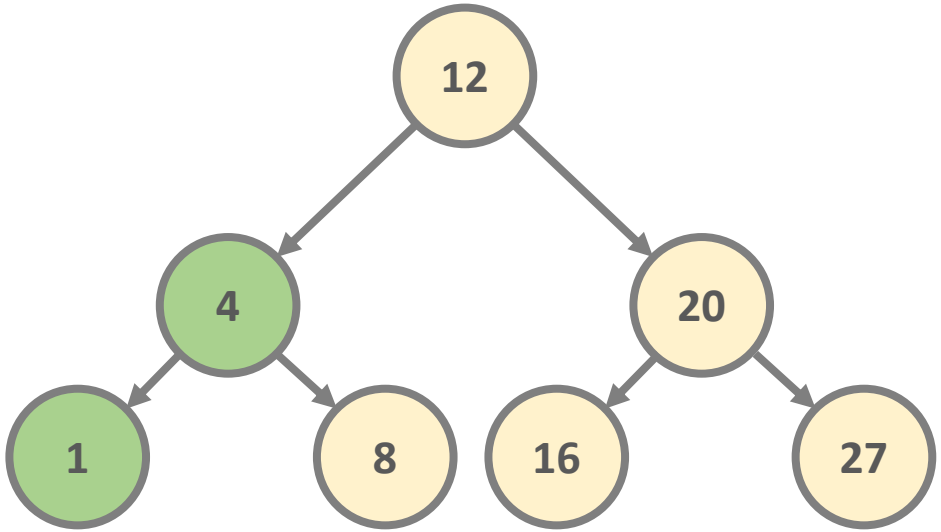



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

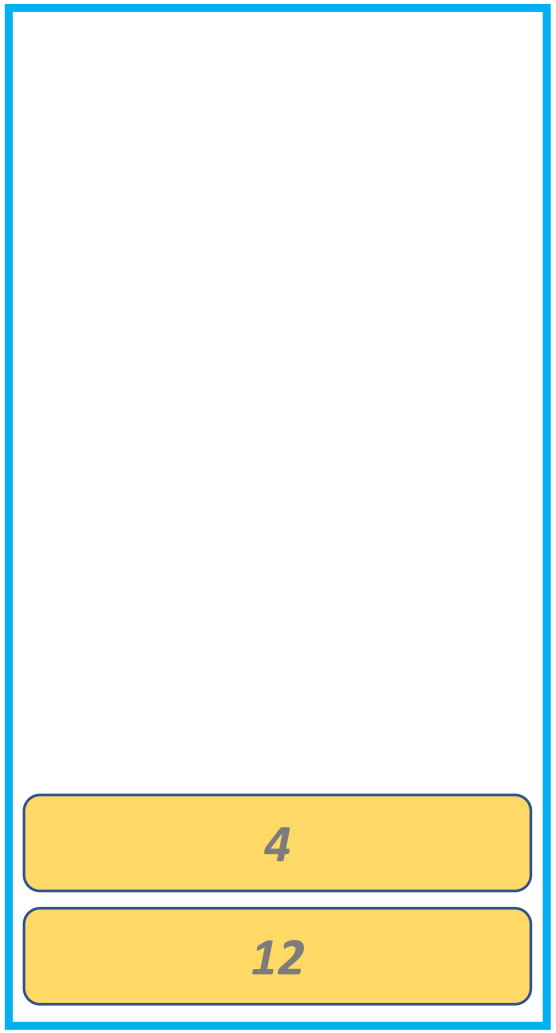


STACK

Binary Search Trees

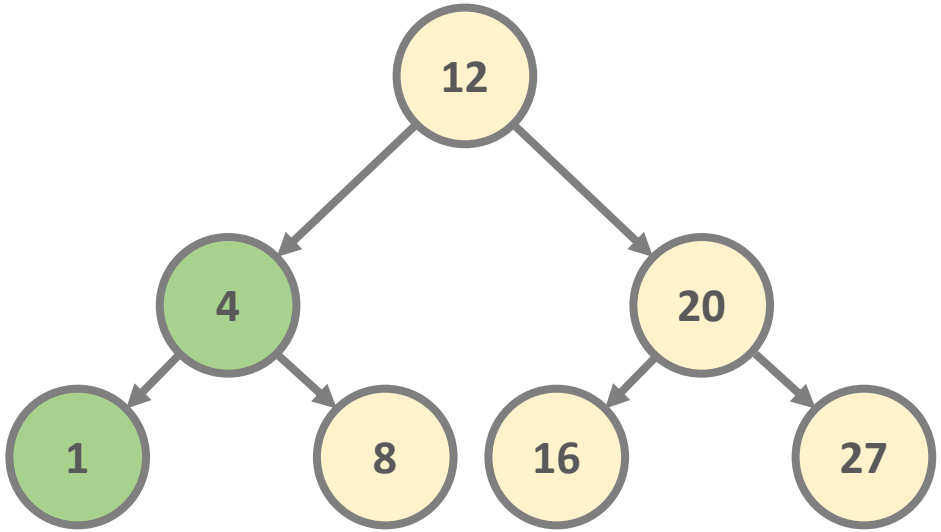



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

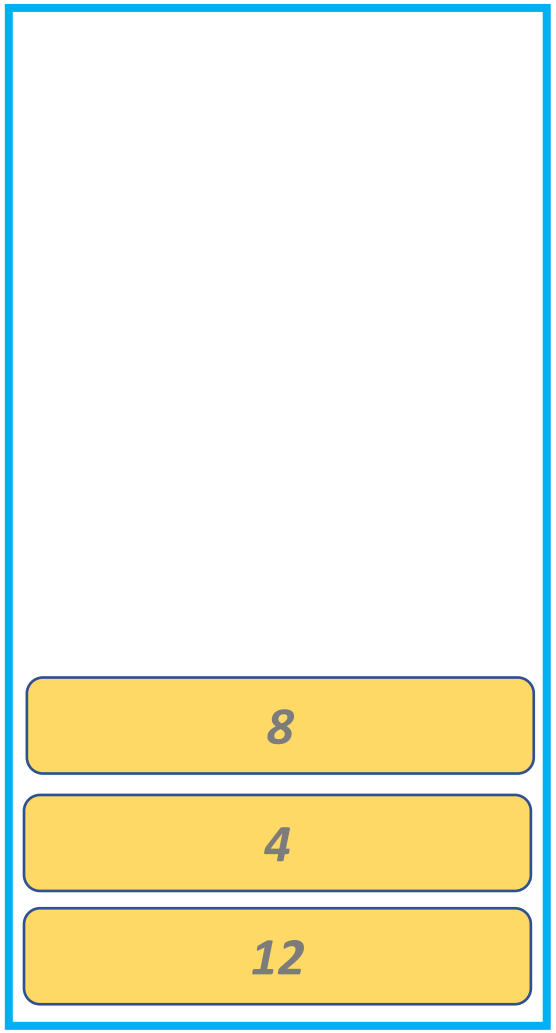


STACK

Binary Search Trees

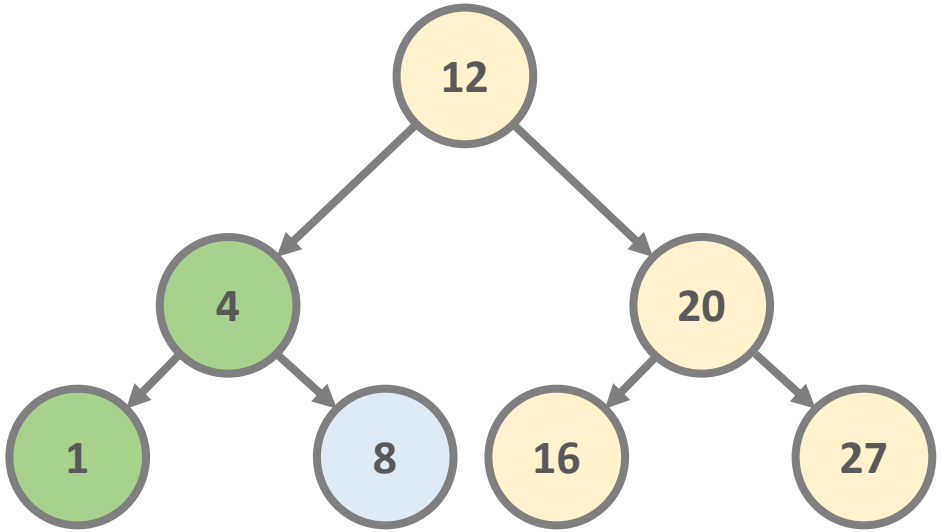



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

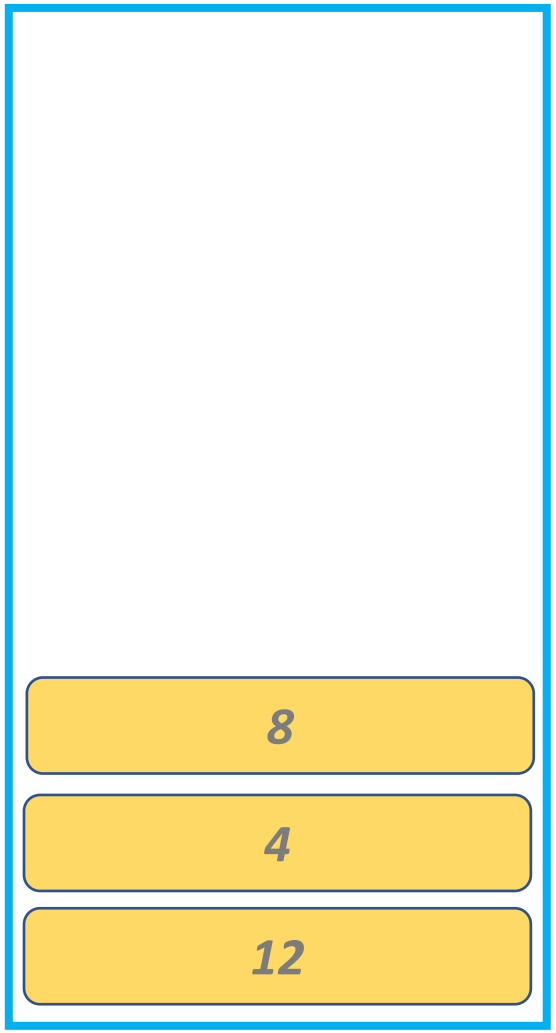


traverse(node):

if node left child is not NULL:
traverse(node left child)

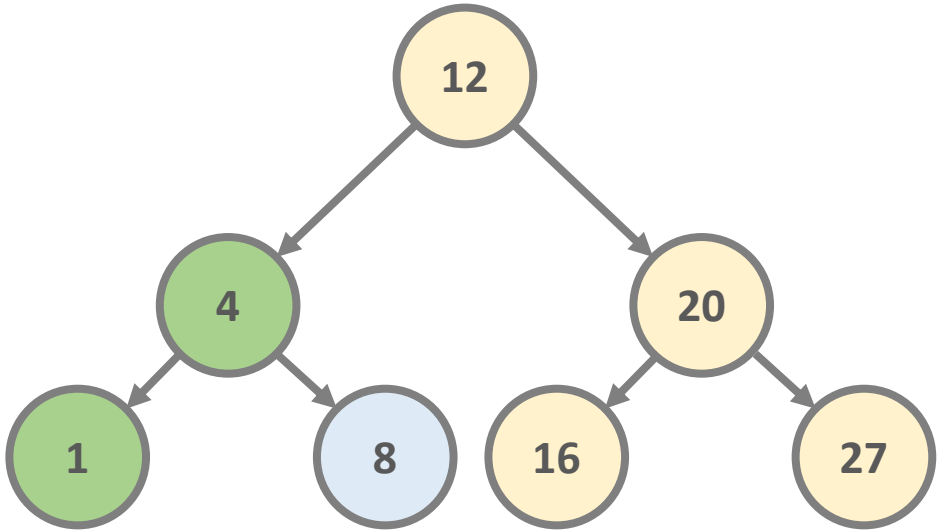
print node.data

if node right child is not NULL:
traverse(node right child)

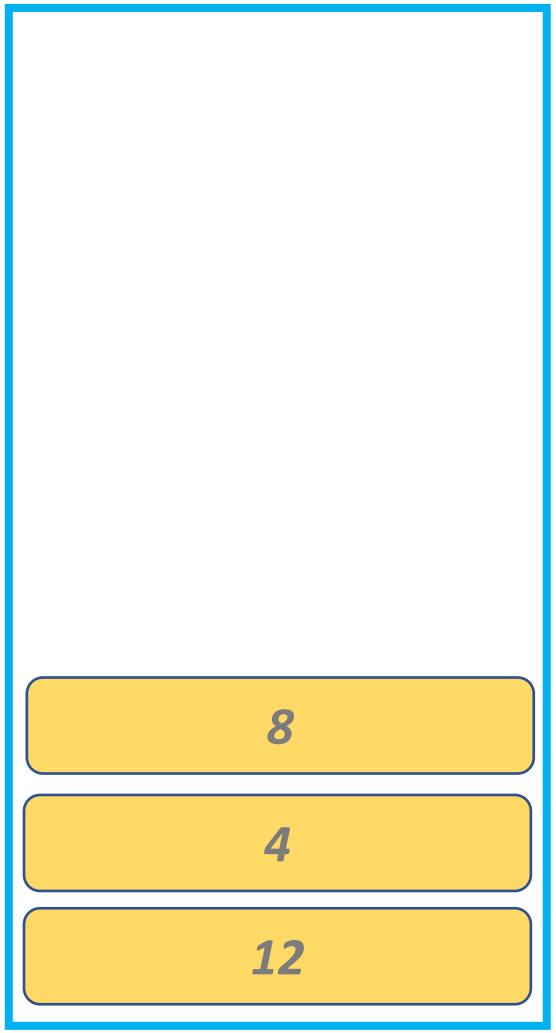


STACK

Binary Search Trees

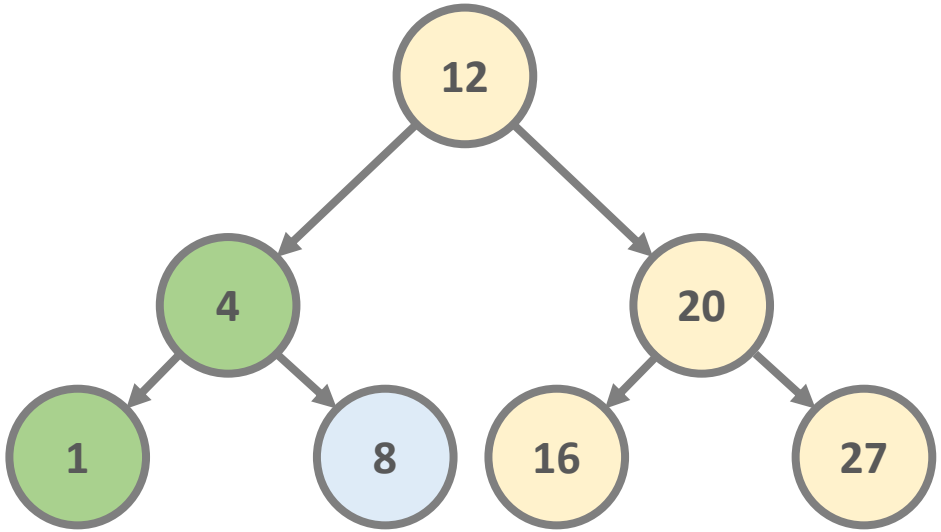



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

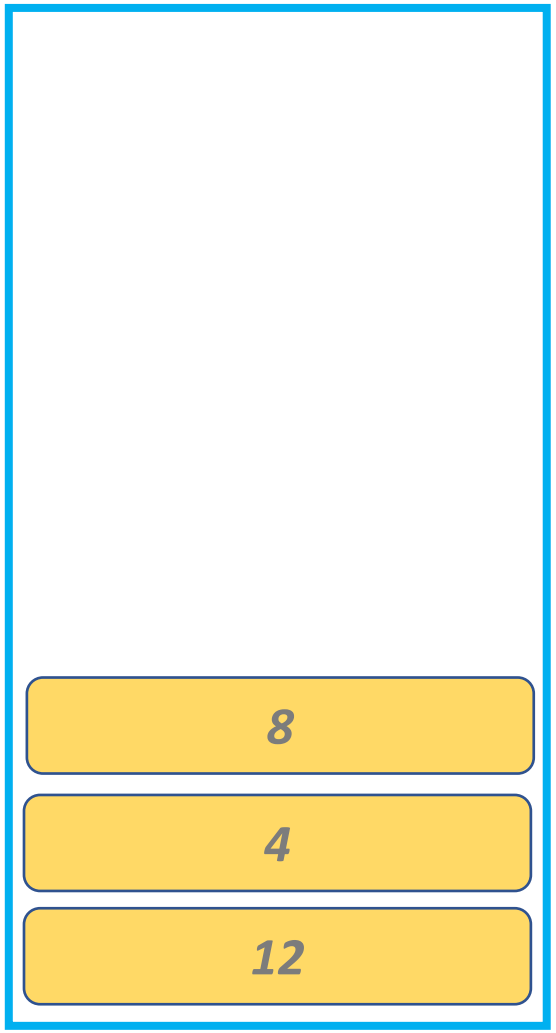


STACK

Binary Search Trees

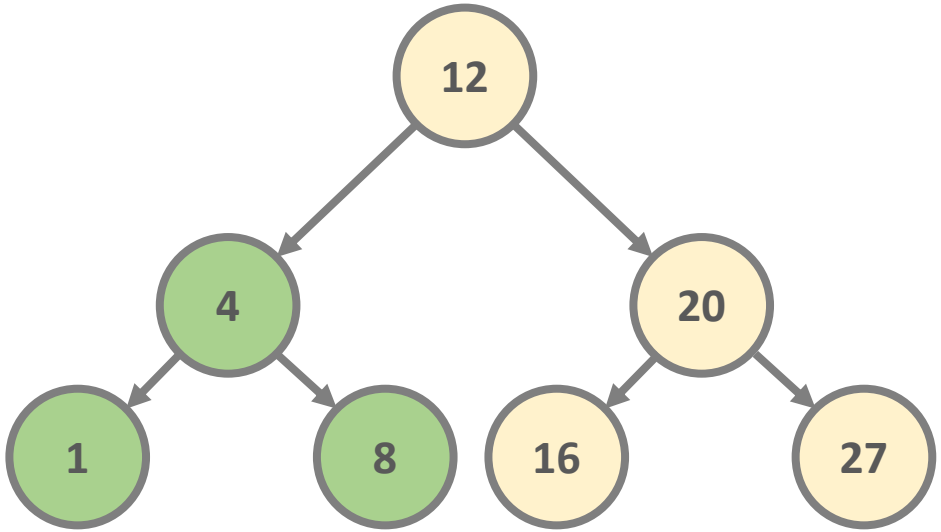



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

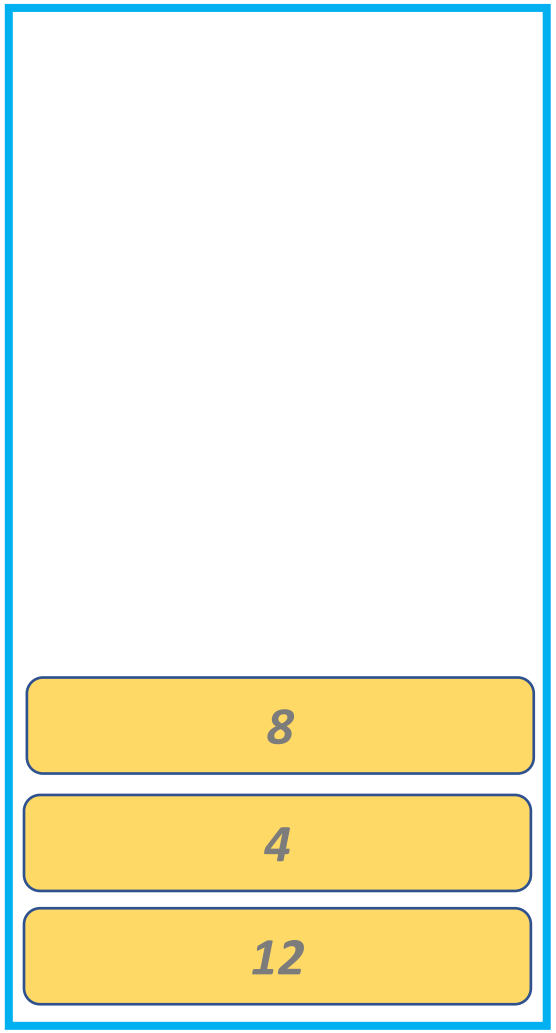


STACK

Binary Search Trees

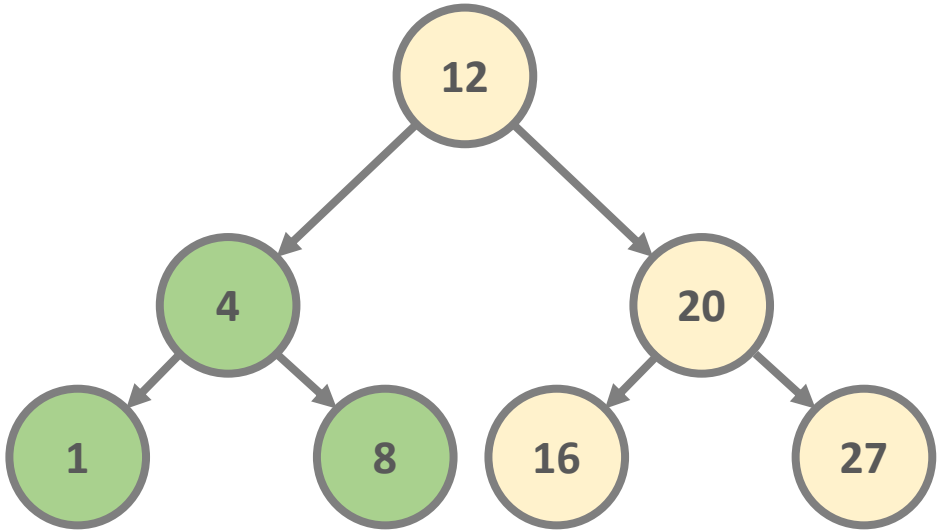



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

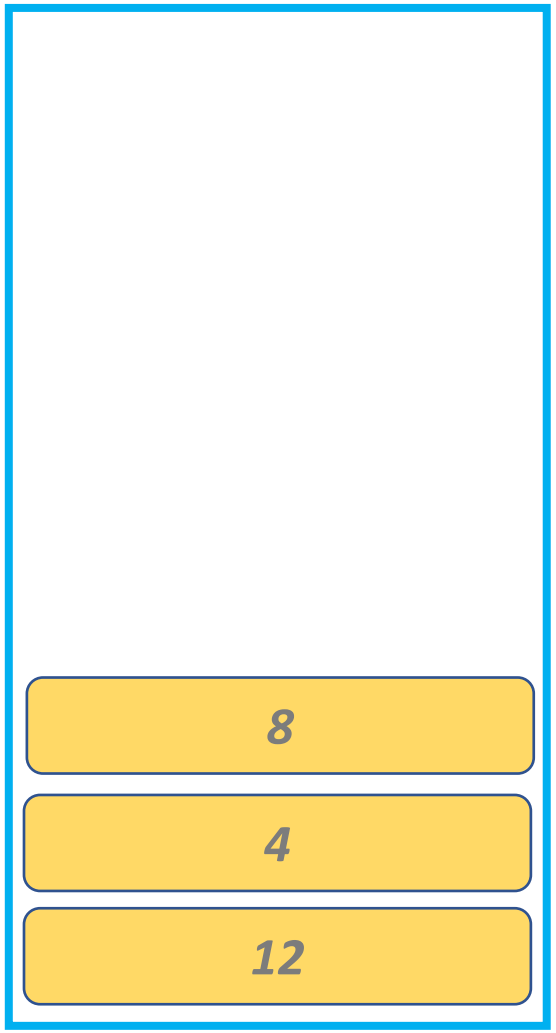


STACK

Binary Search Trees

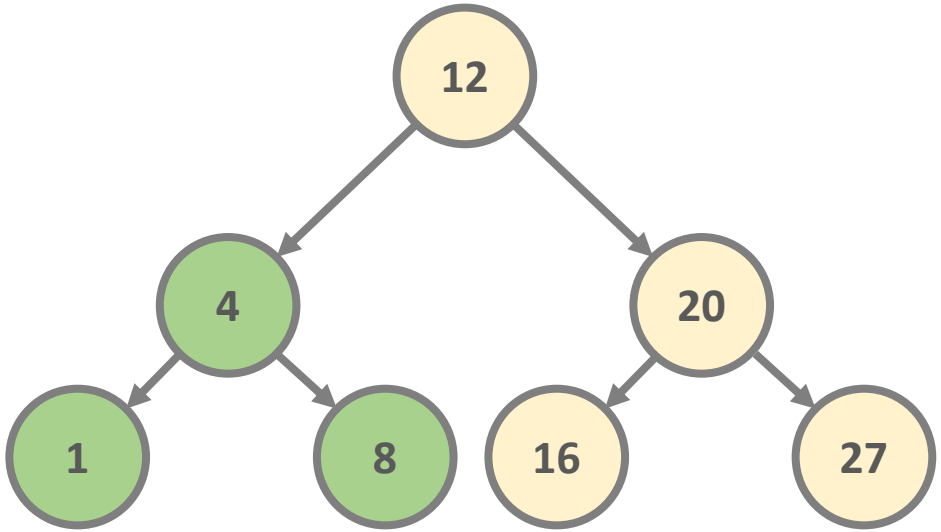



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

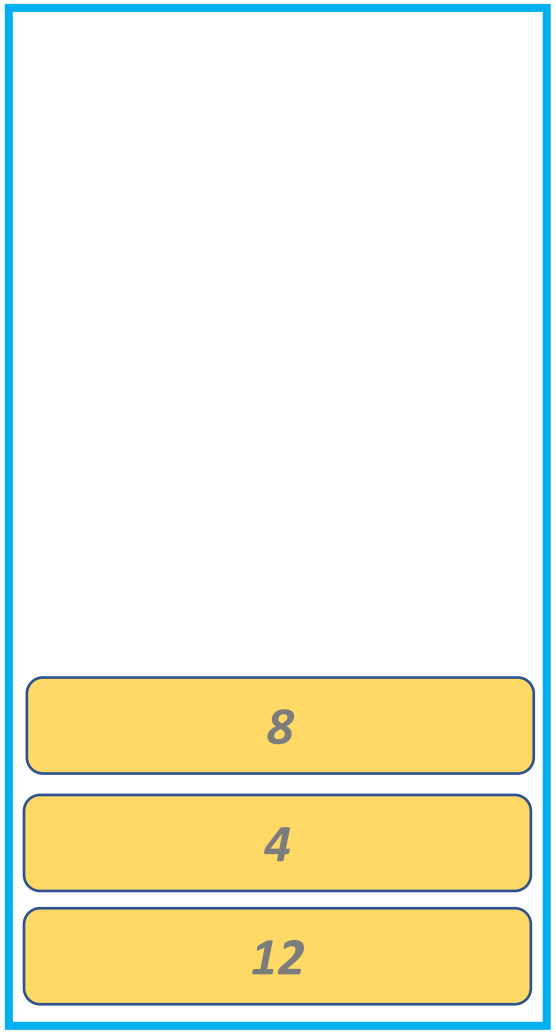


STACK

Binary Search Trees

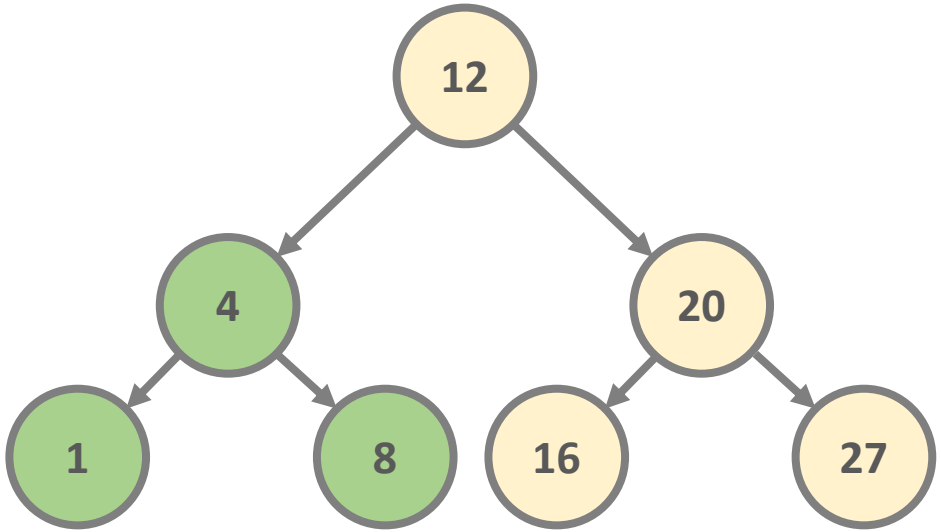



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

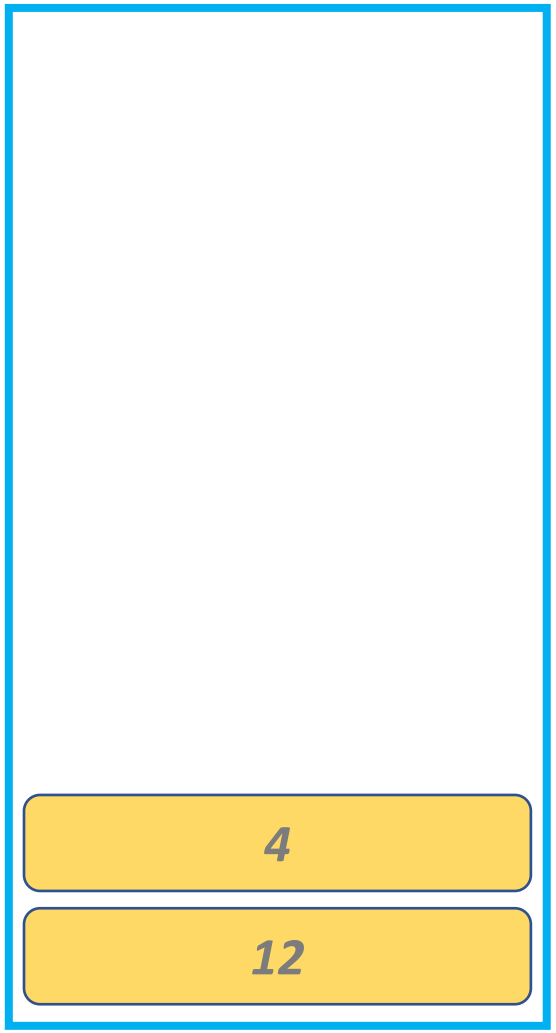


STACK

Binary Search Trees

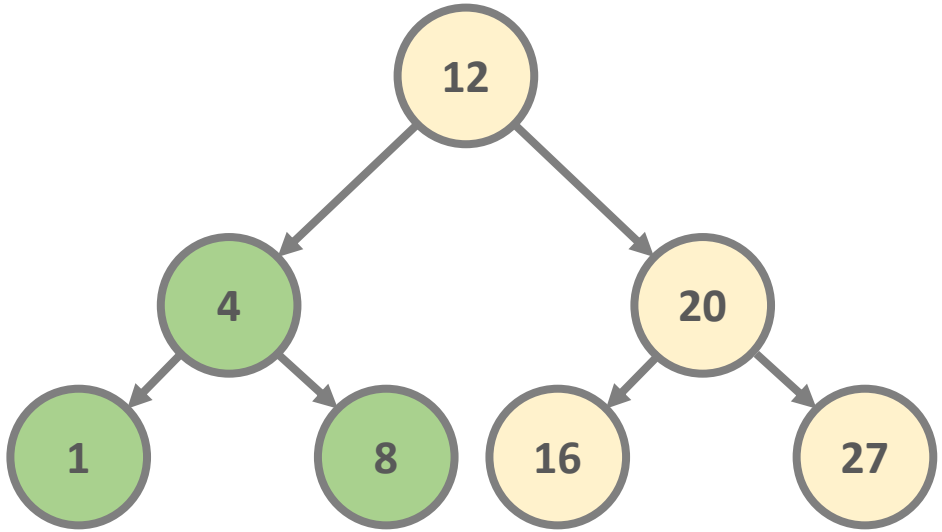



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

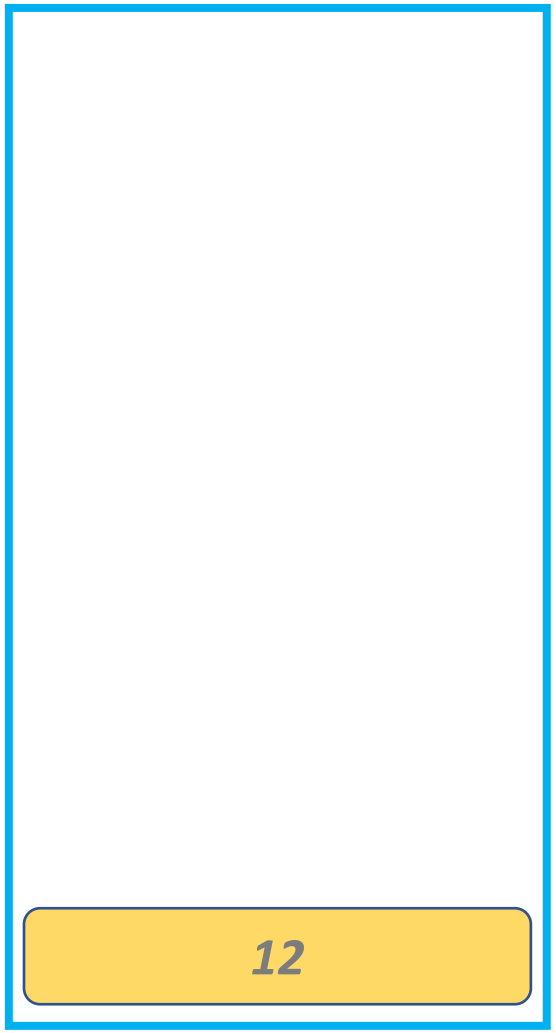


STACK

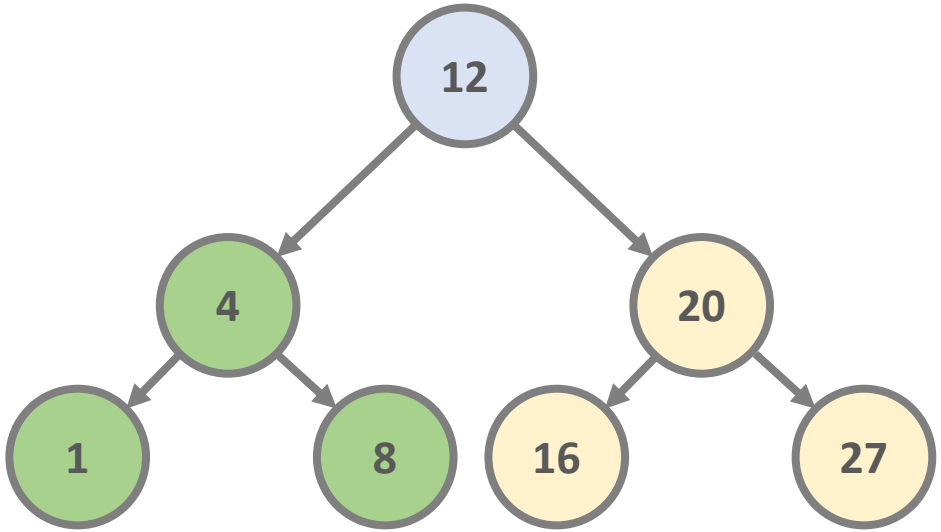
Binary Search Trees



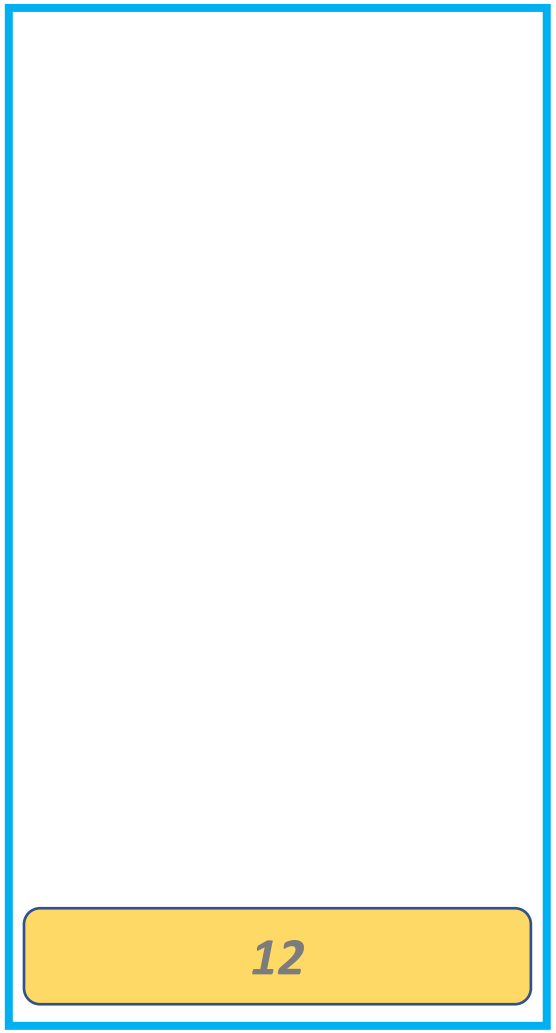

```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



Binary Search Trees

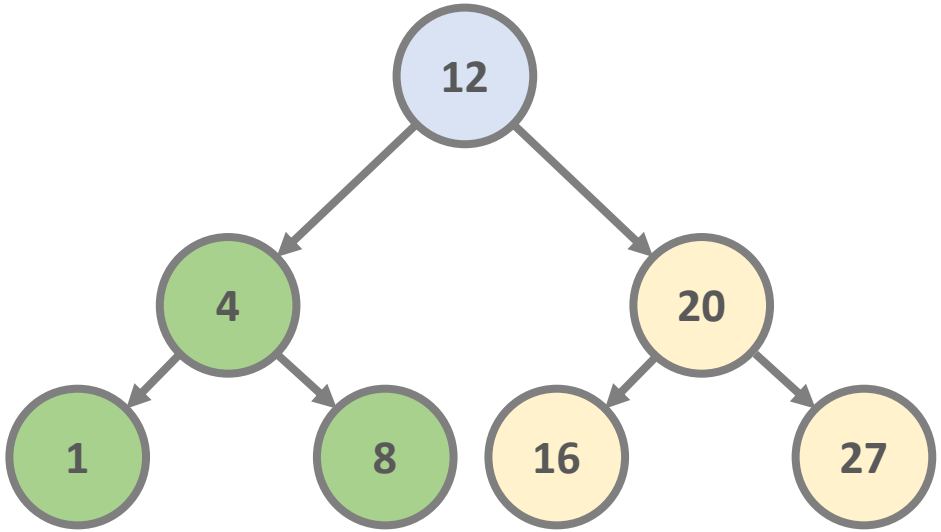



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

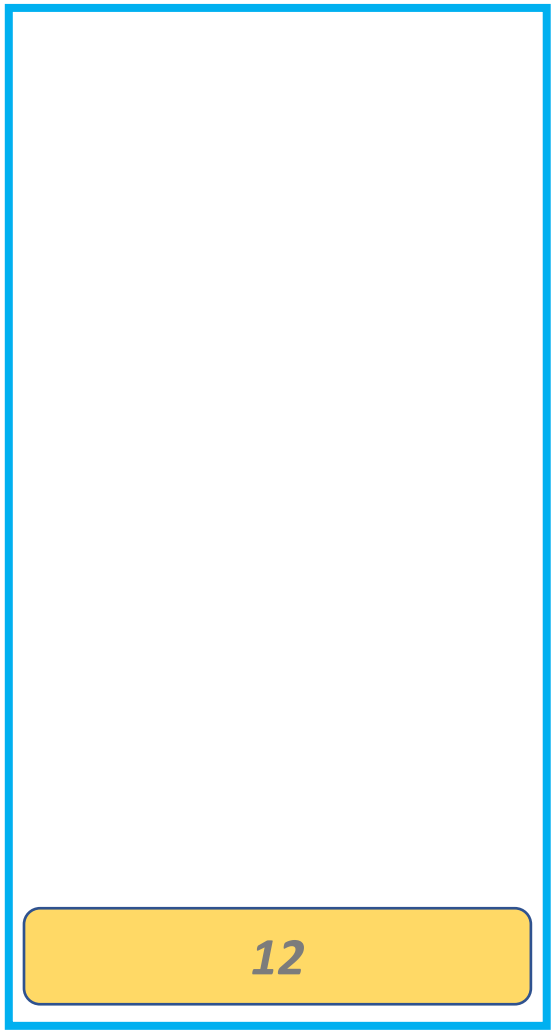


STACK

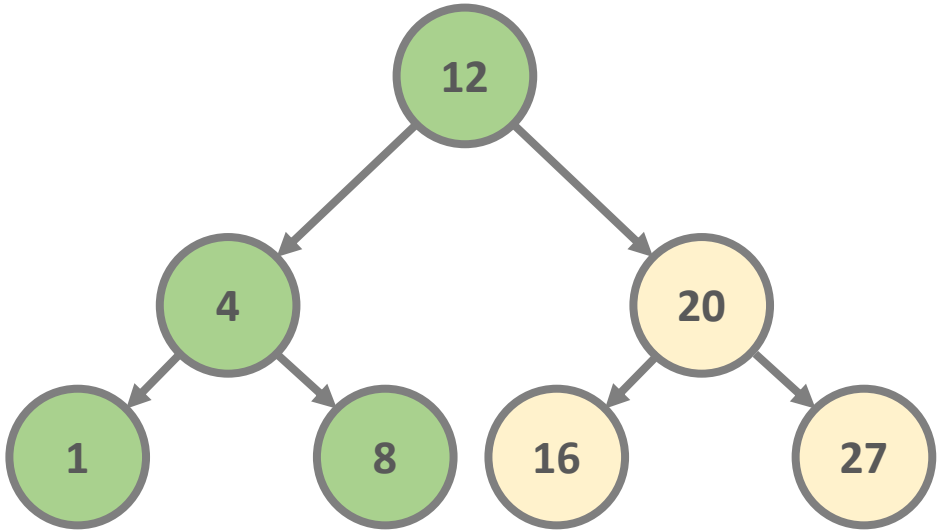
Binary Search Trees



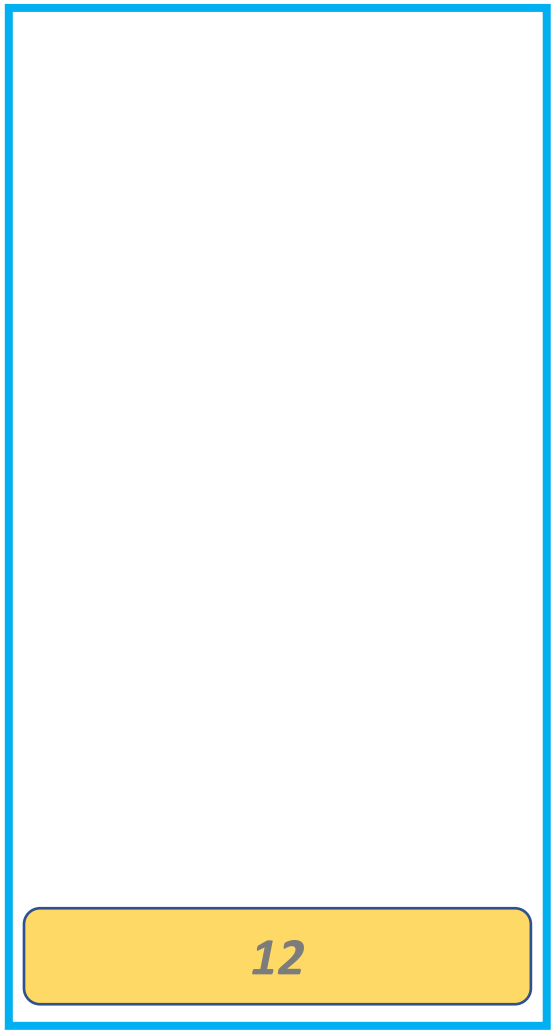

```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



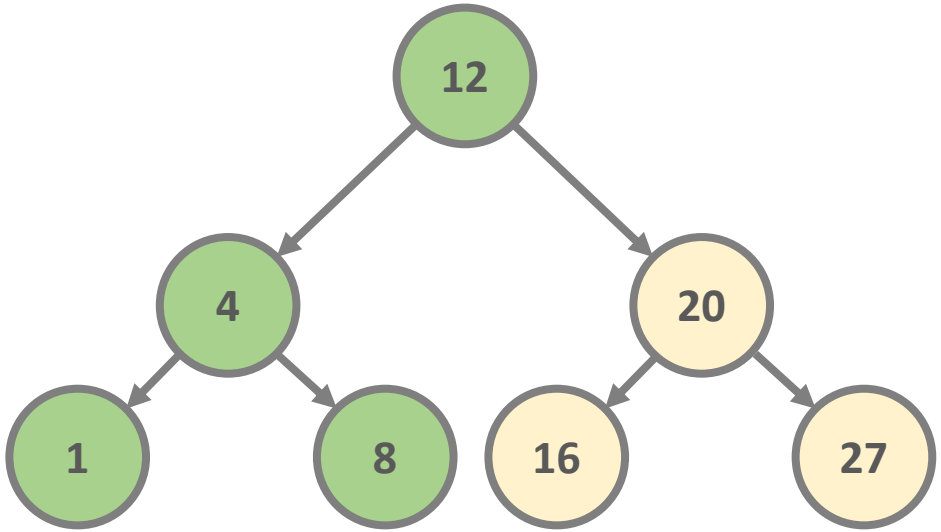
Binary Search Trees



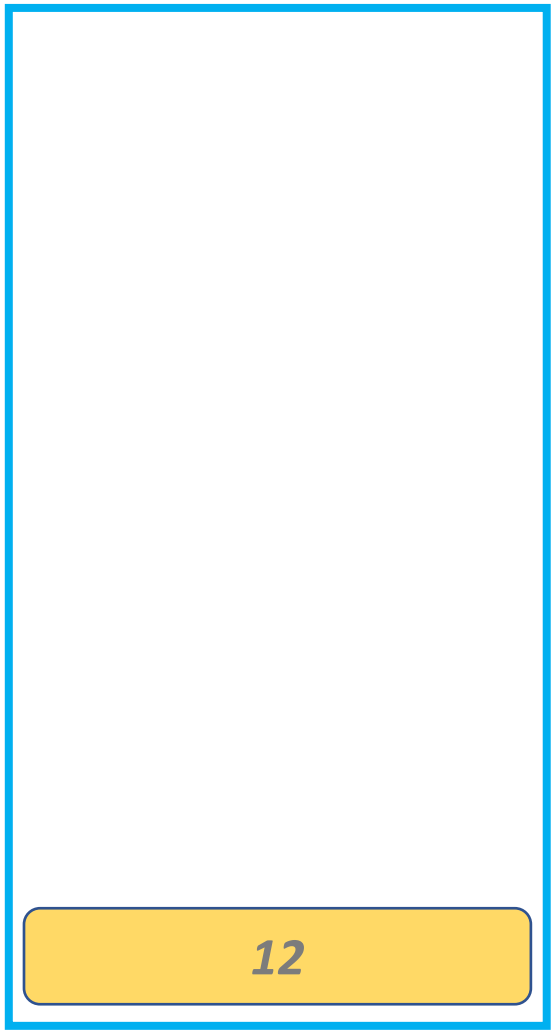

```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



Binary Search Trees

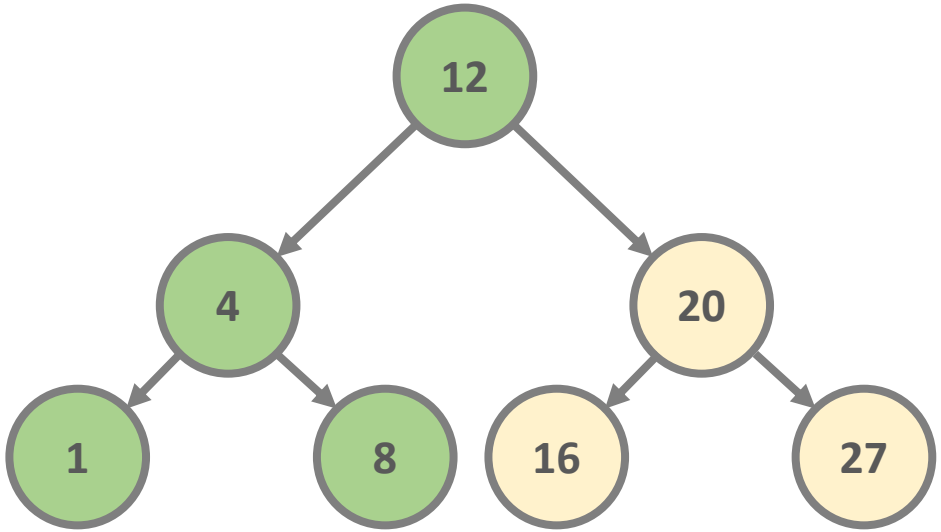



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

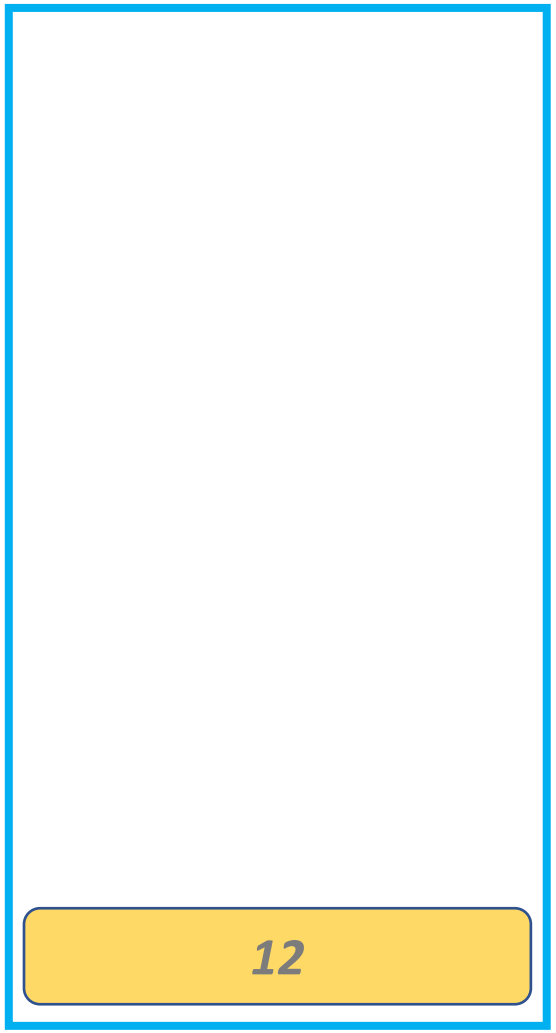


STACK

Binary Search Trees

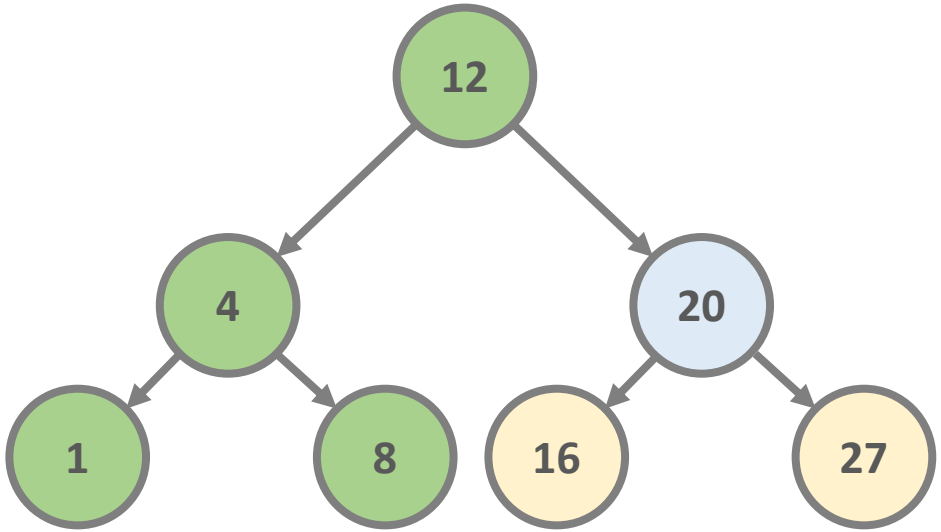



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

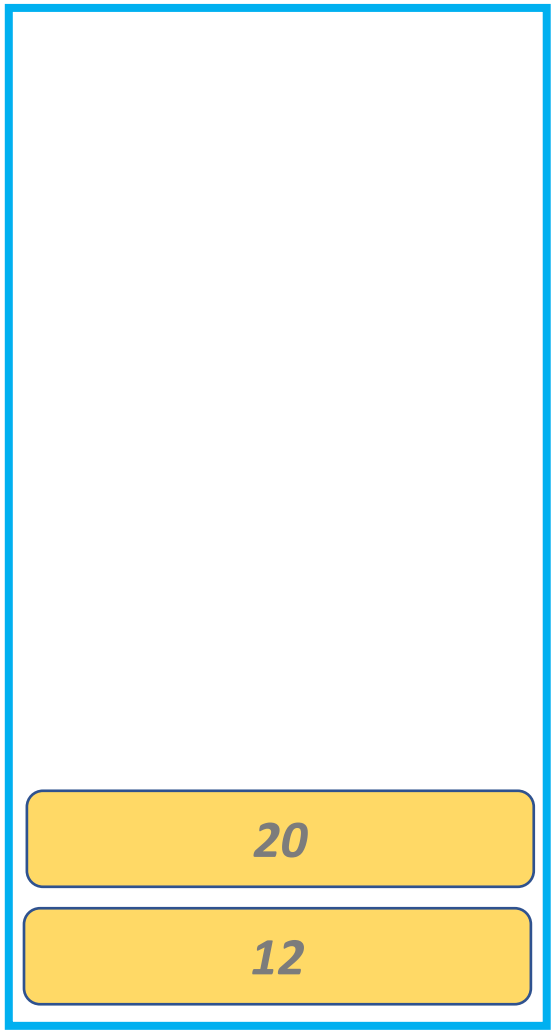


STACK

Binary Search Trees

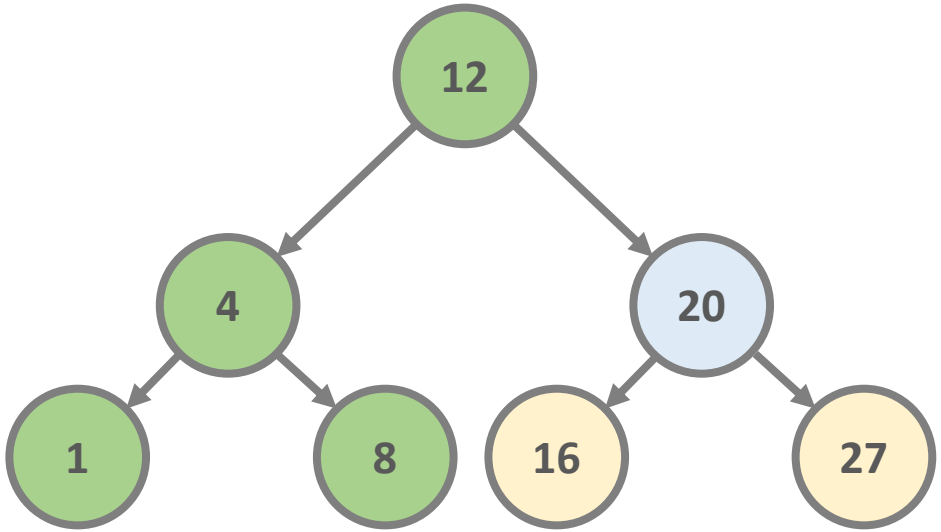



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

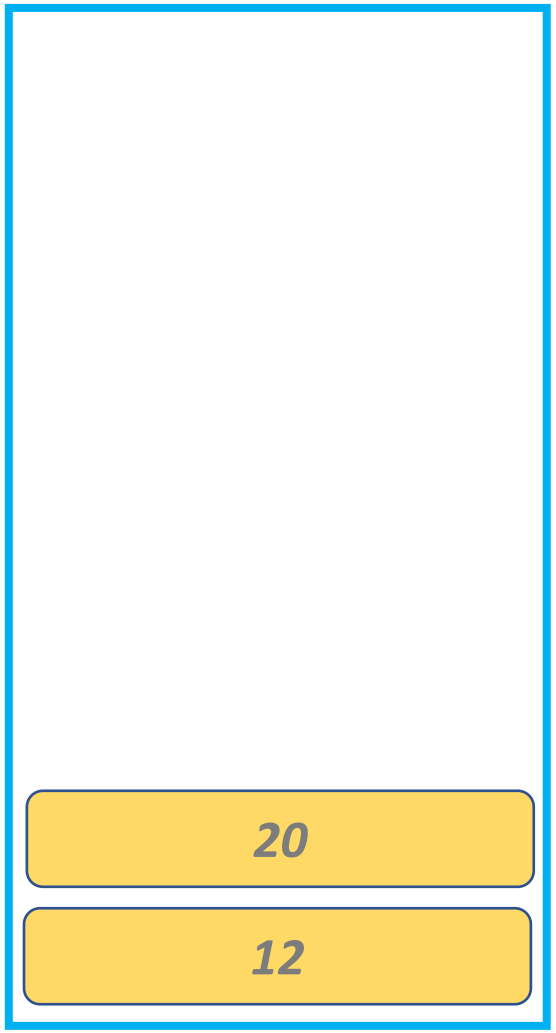


STACK

Binary Search Trees

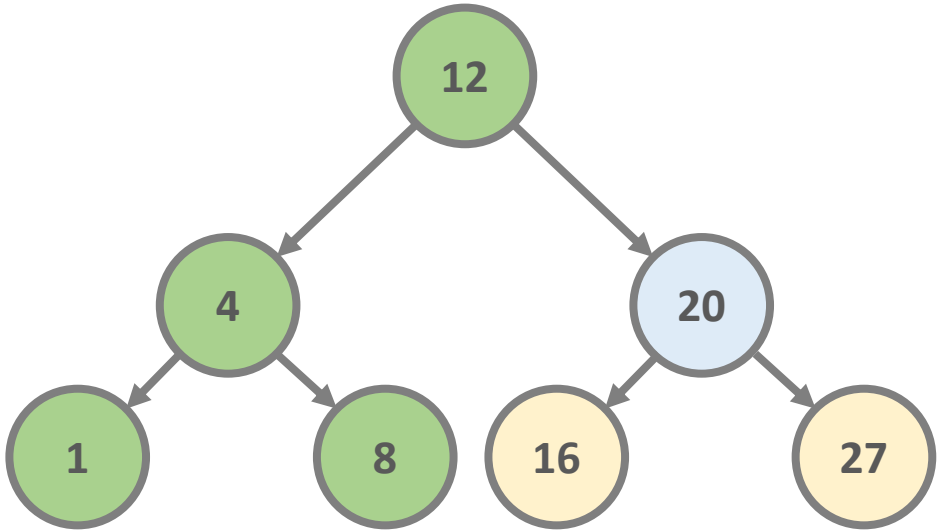



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

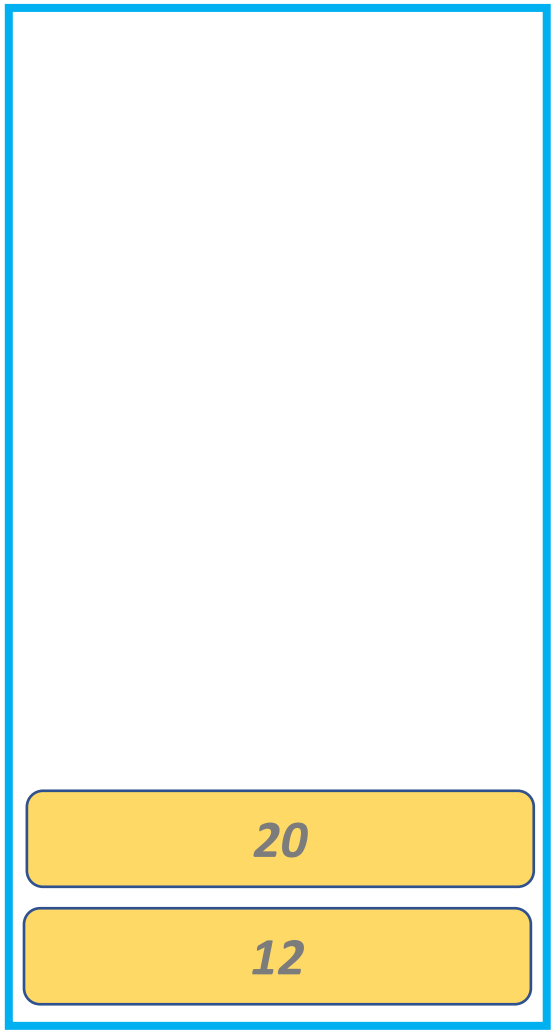


STACK

Binary Search Trees

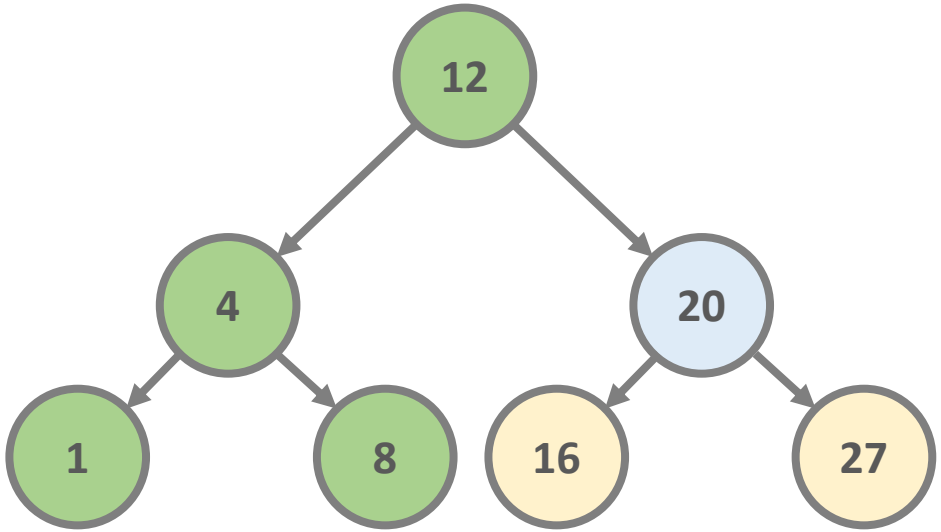



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

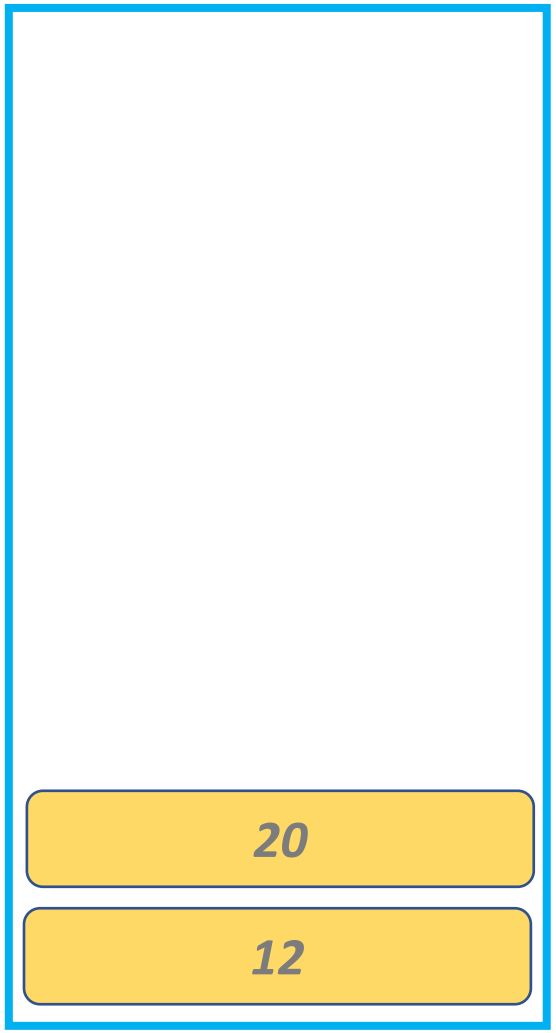


STACK

Binary Search Trees

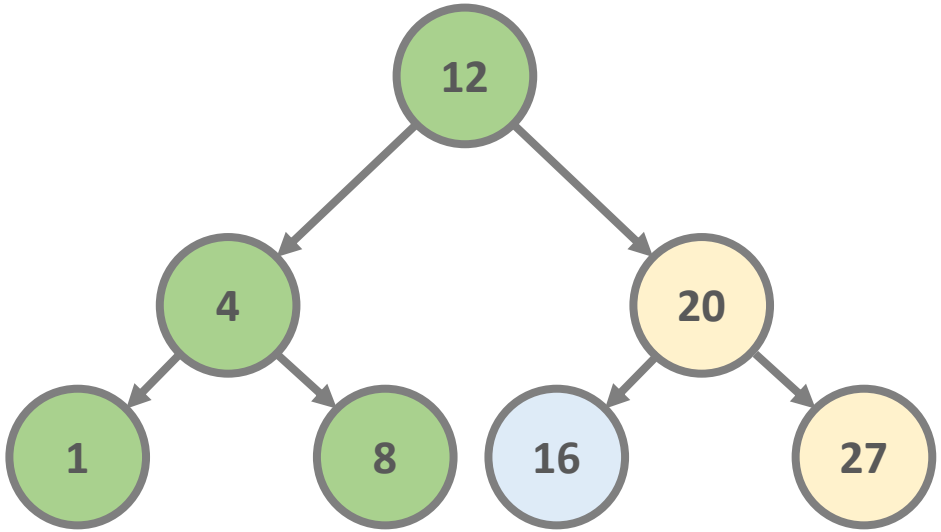



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

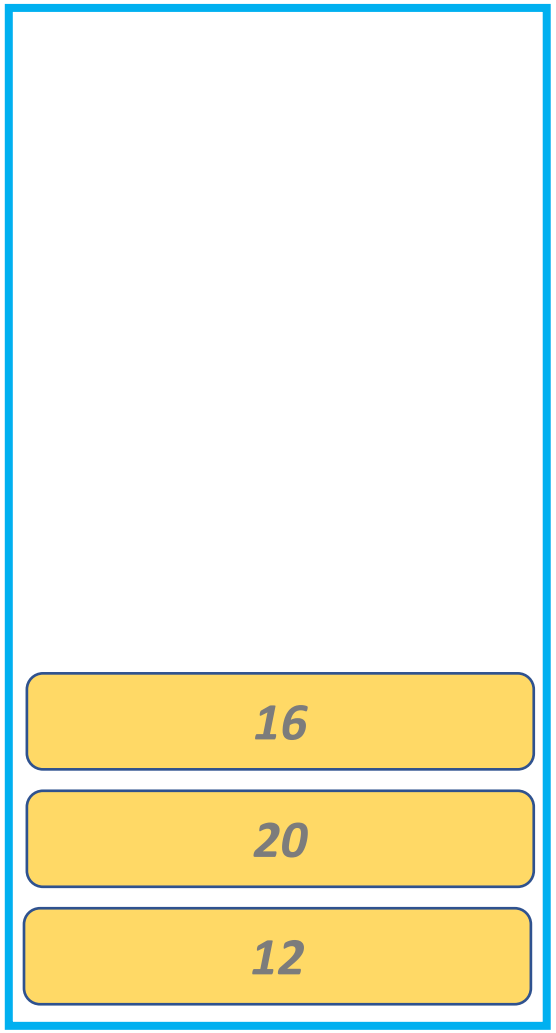


STACK

Binary Search Trees

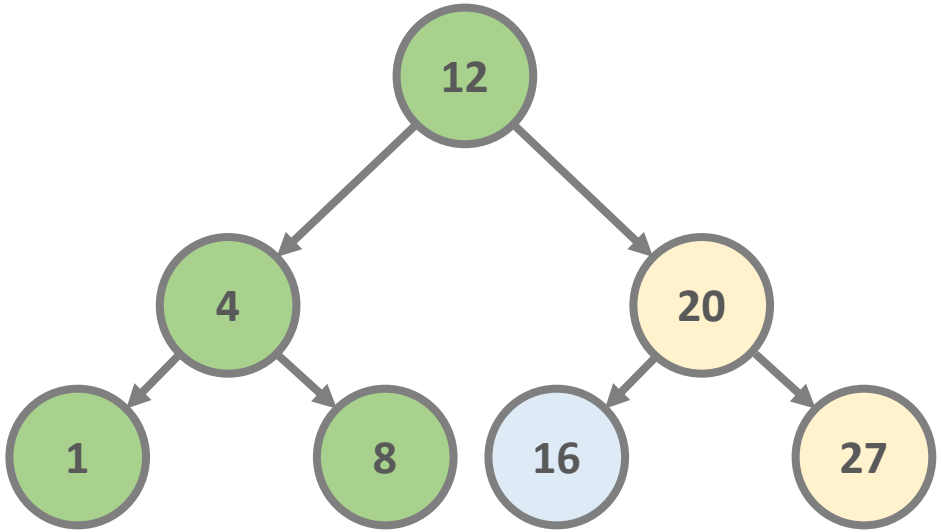



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

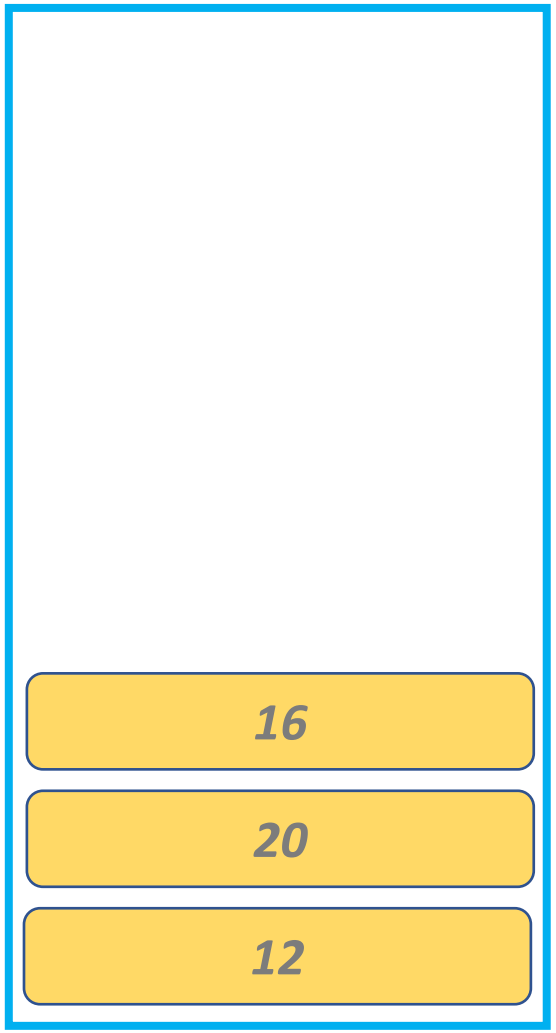


traverse(node):

if node left child is not NULL:
traverse(node left child)

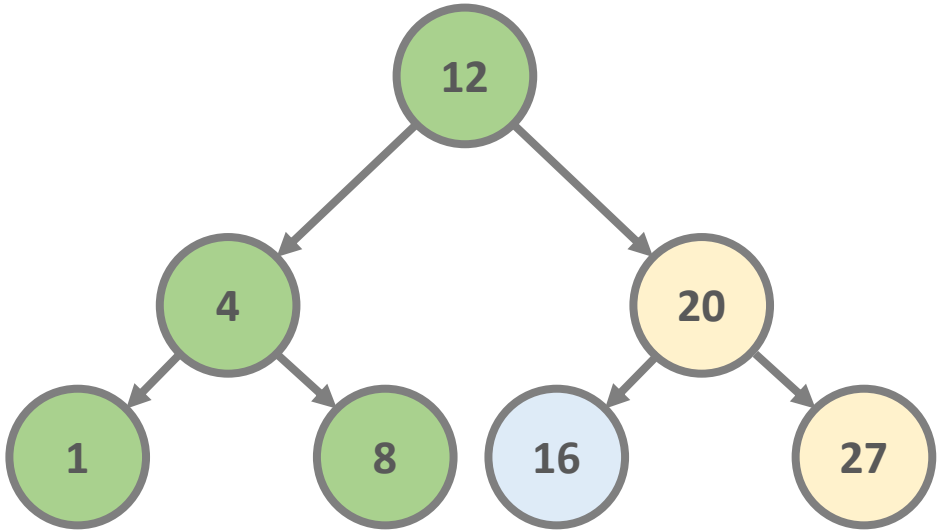
print node.data

if node right child is not NULL:
traverse(node right child)

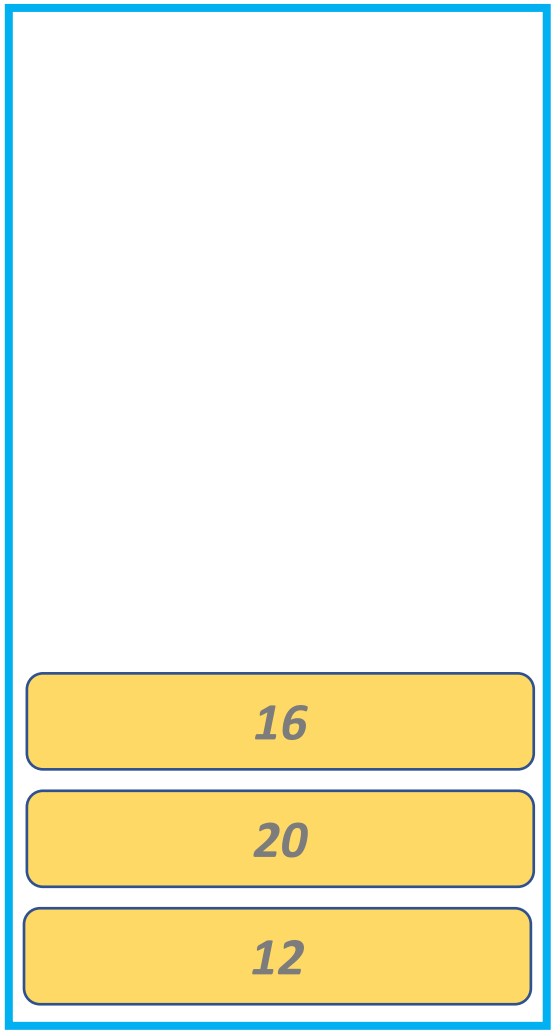


STACK

Binary Search Trees

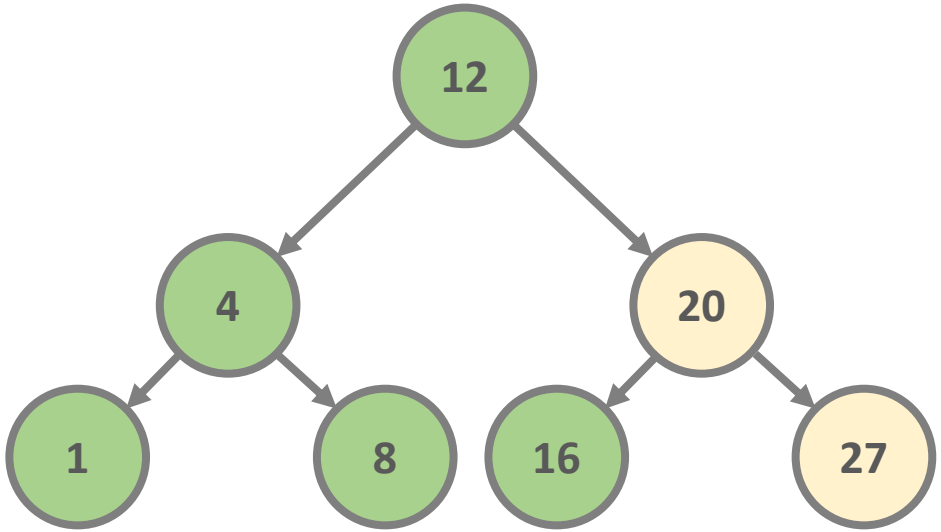



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

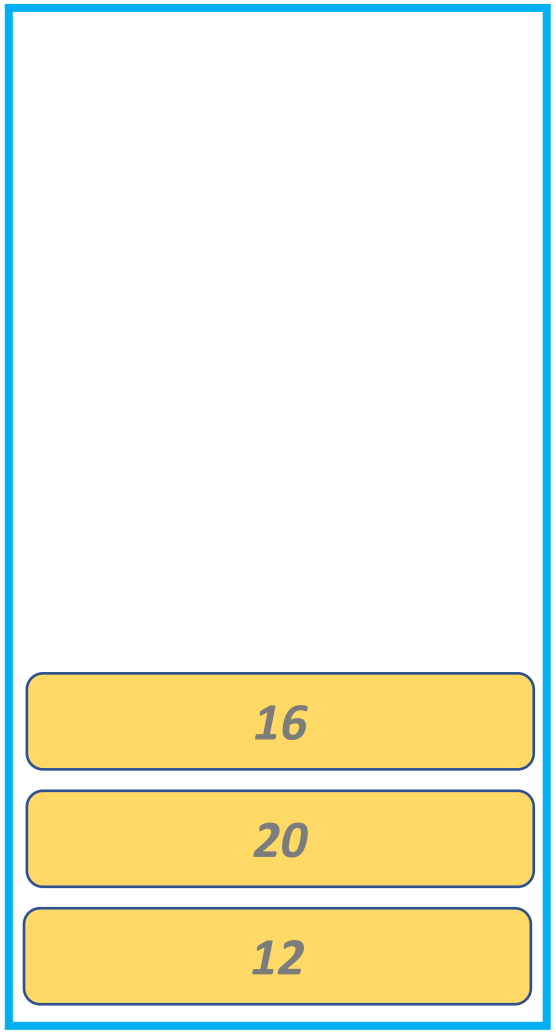


traverse(node):

if node left child is not NULL:
traverse(node left child)

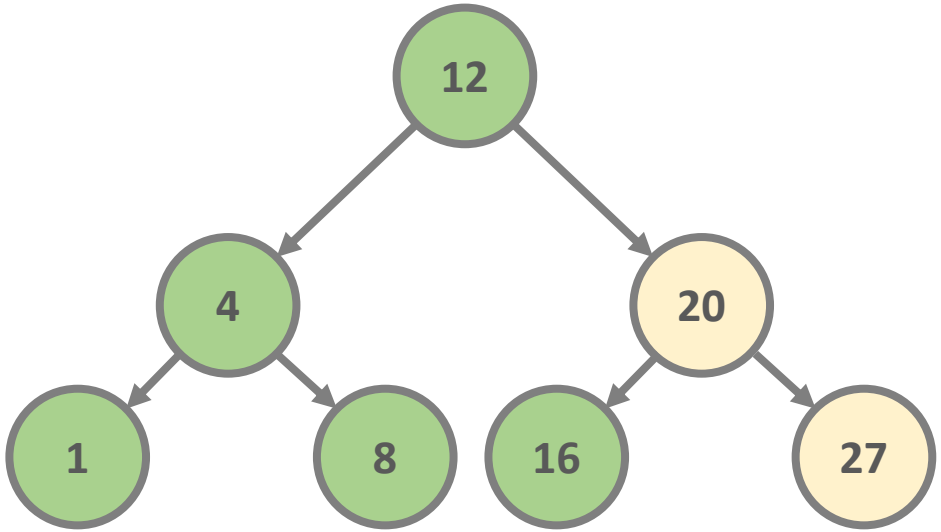
print node.data

if node right child is not NULL:
traverse(node right child)

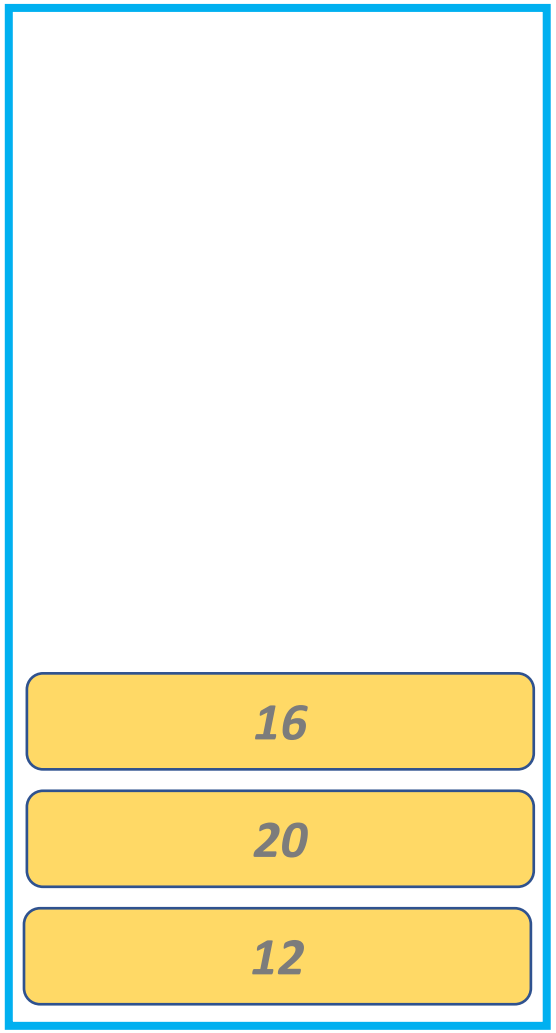


STACK

Binary Search Trees

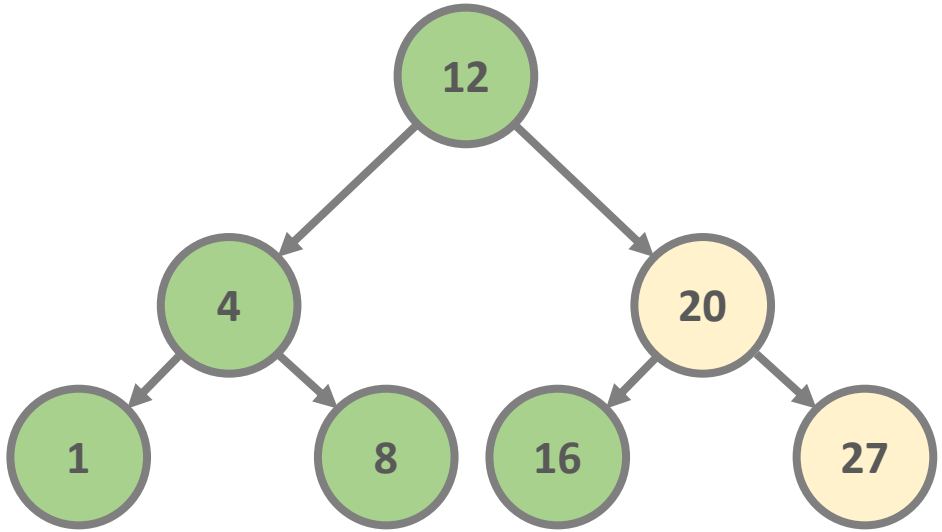



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

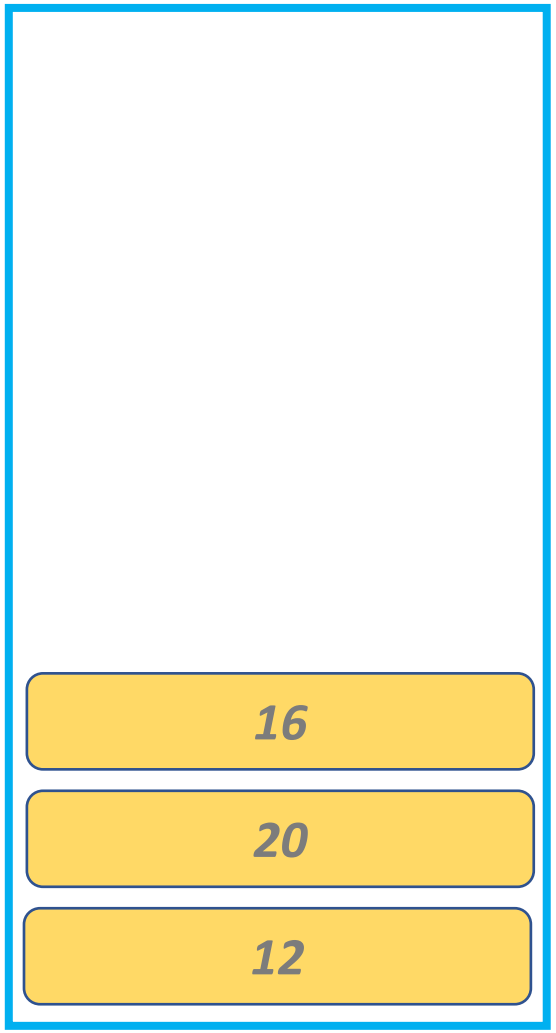


traverse(node):

if node left child is not NULL:
traverse(node left child)

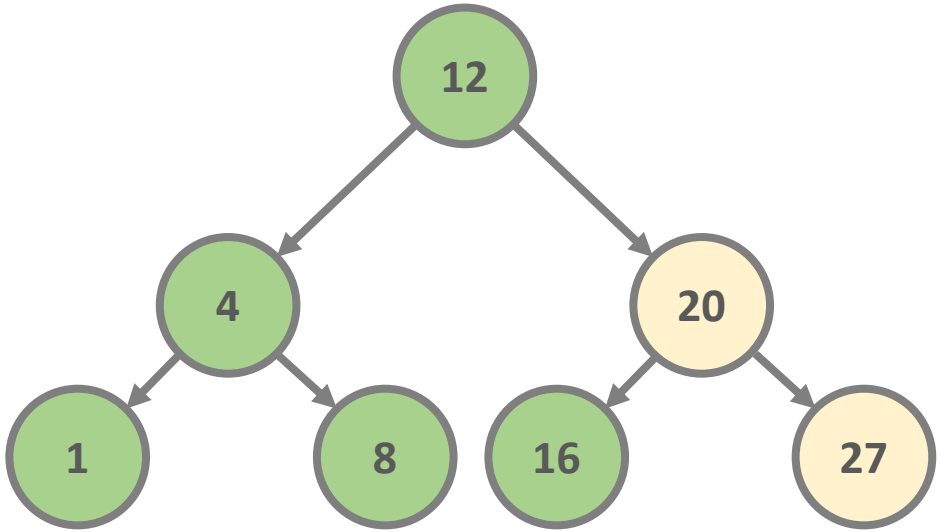
print node.data

if node right child is not NULL:
traverse(node right child)

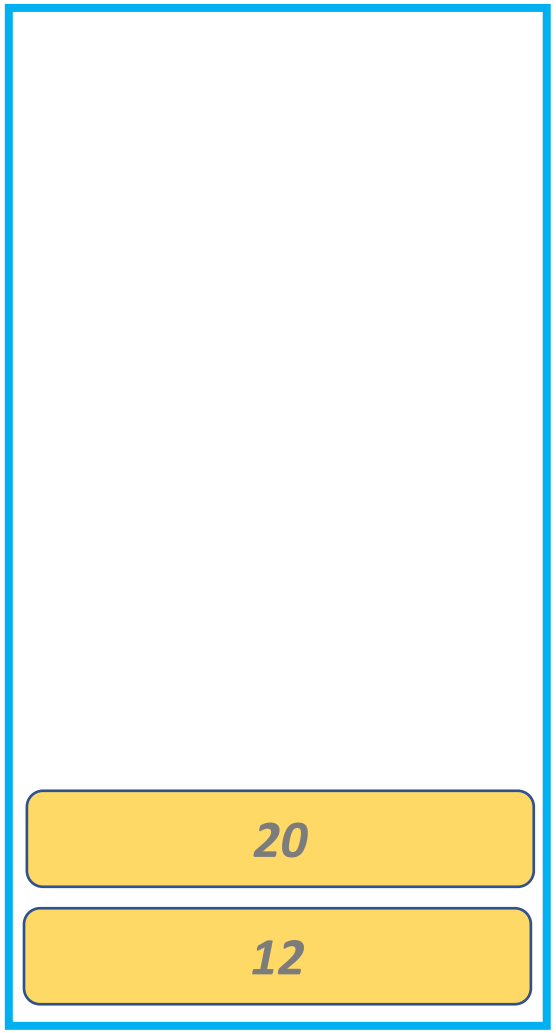


STACK

Binary Search Trees

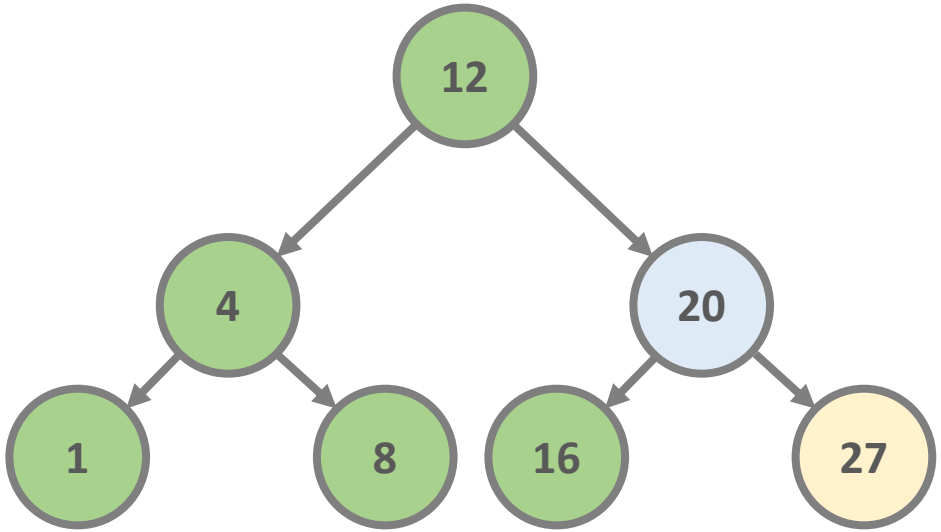



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

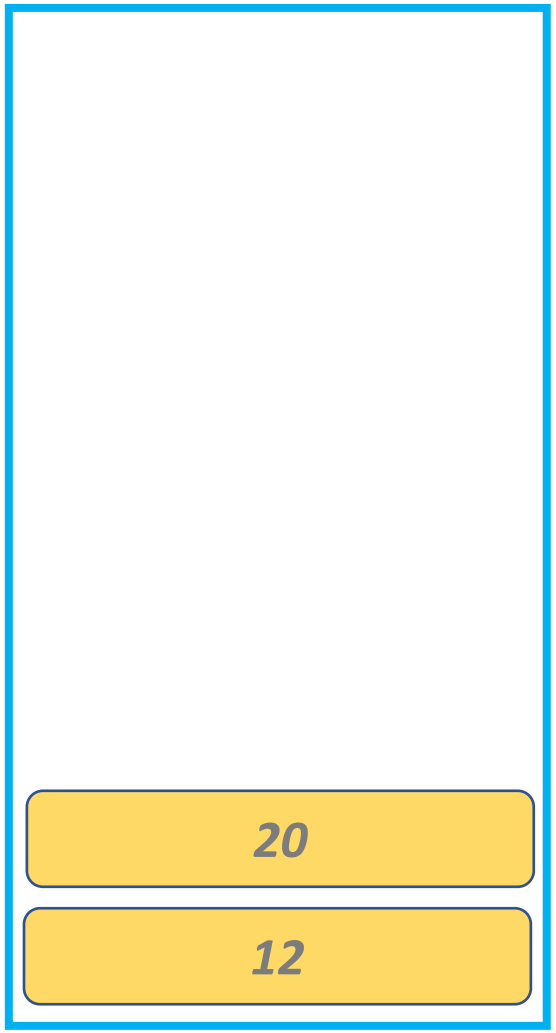


STACK

Binary Search Trees

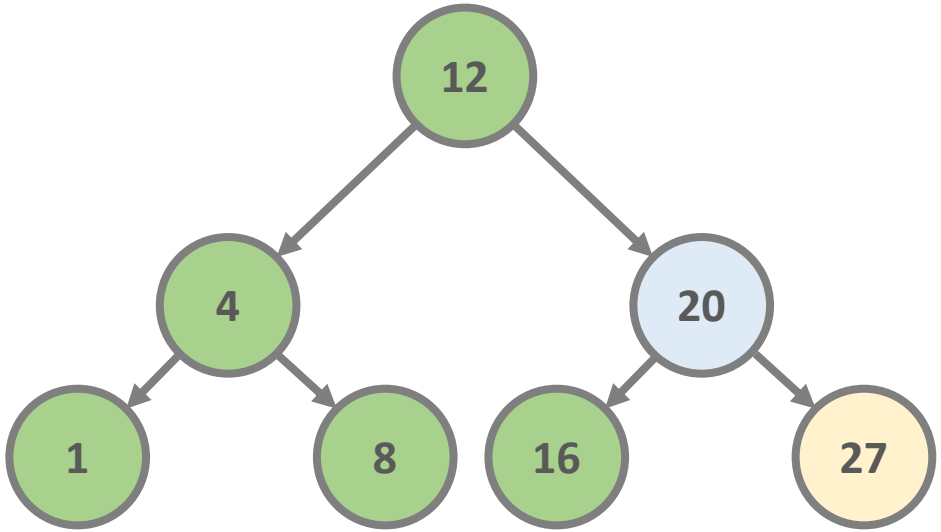



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

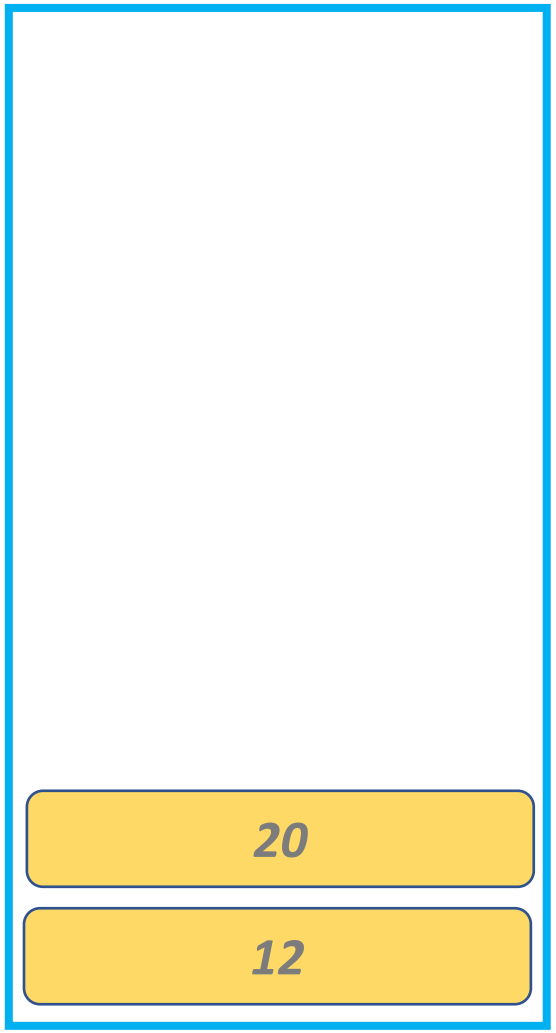


STACK

Binary Search Trees

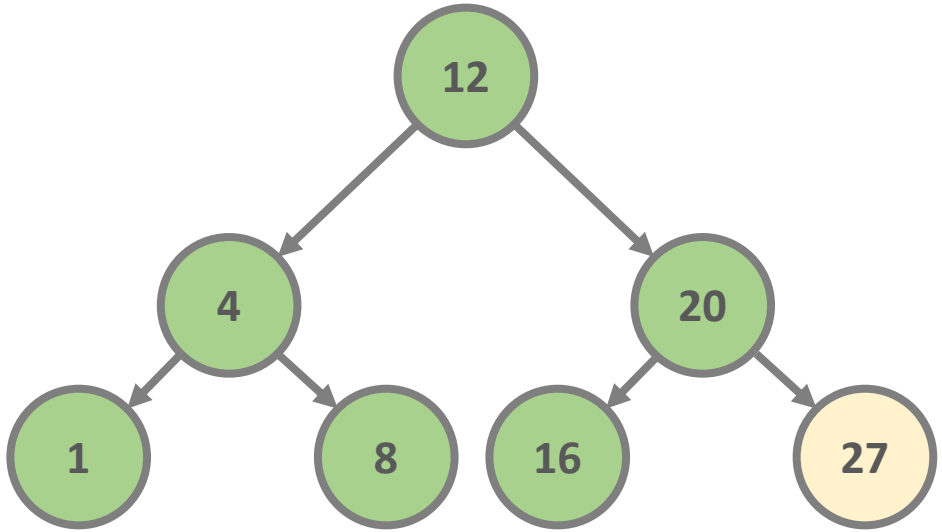



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

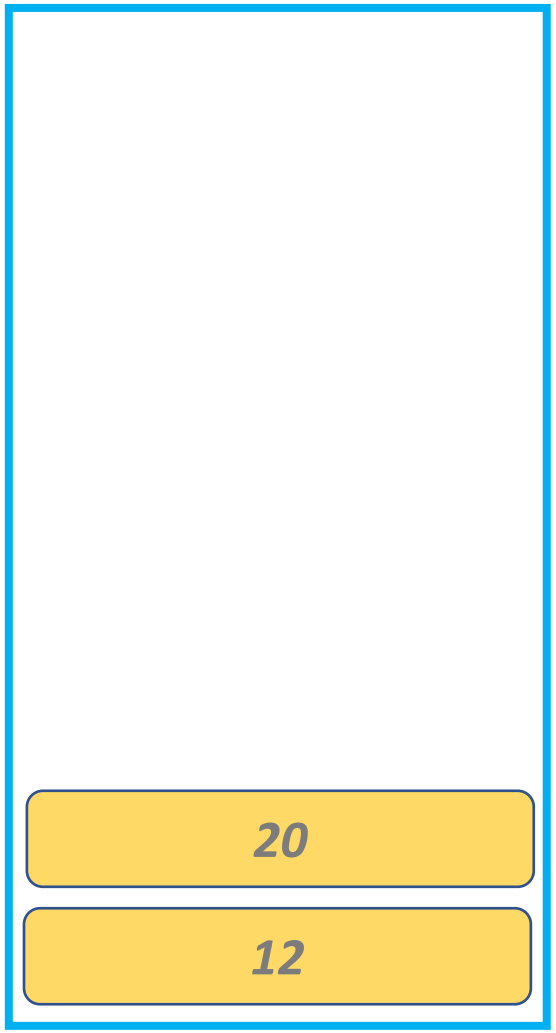


STACK

Binary Search Trees

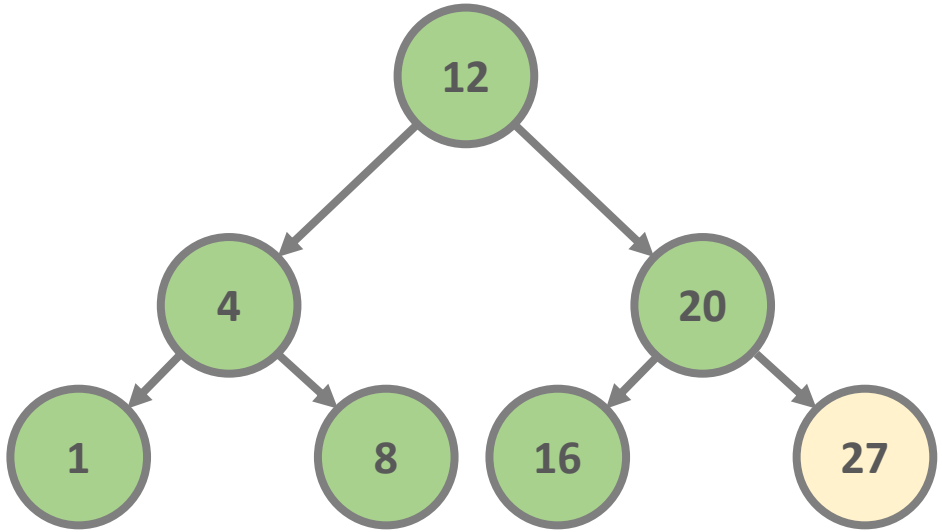



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

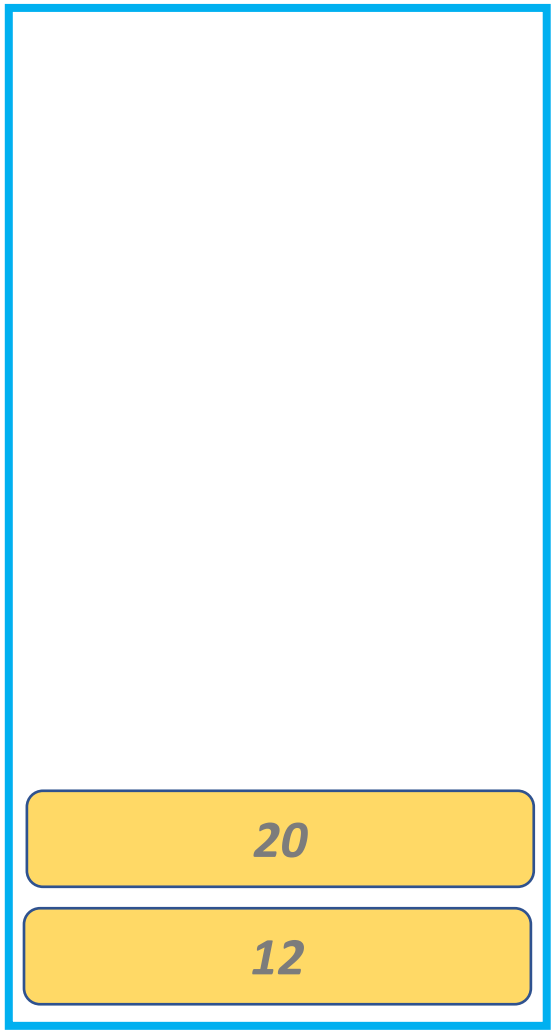


STACK

Binary Search Trees

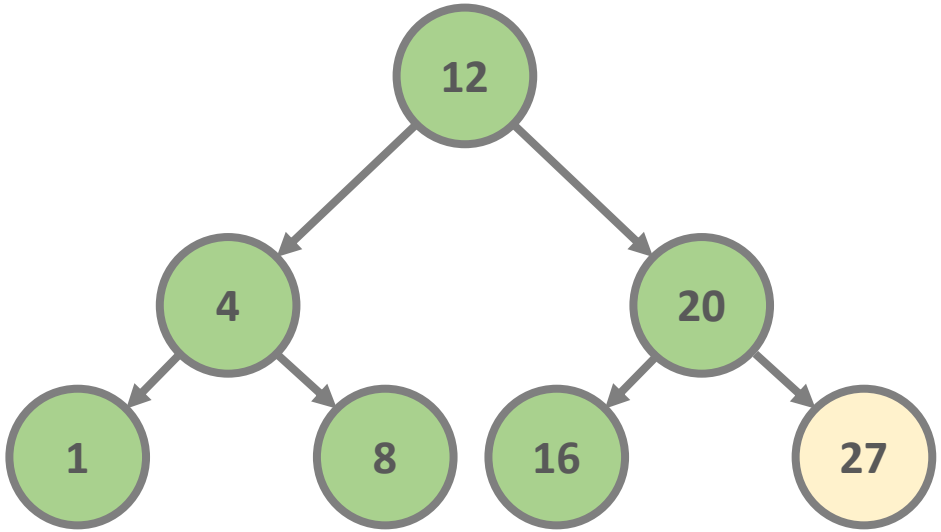



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

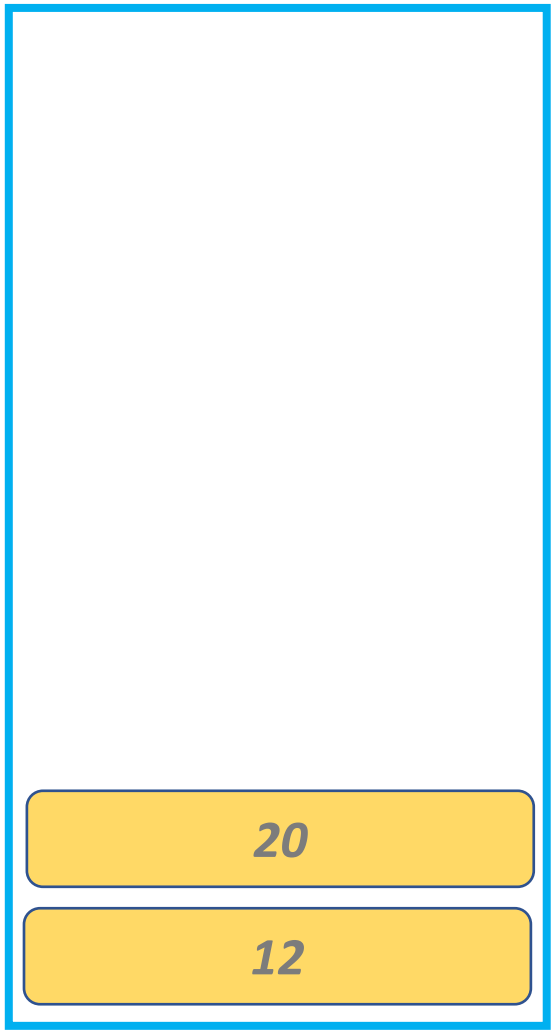


STACK

Binary Search Trees

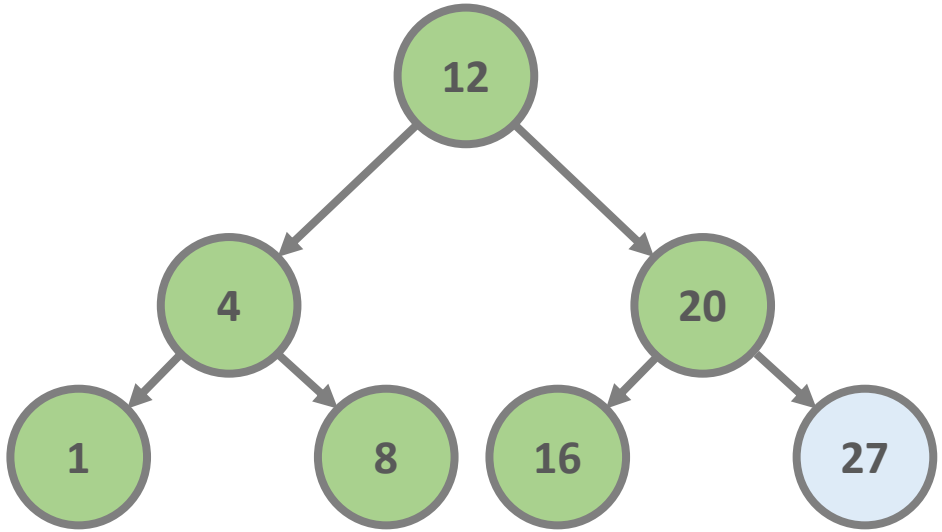



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

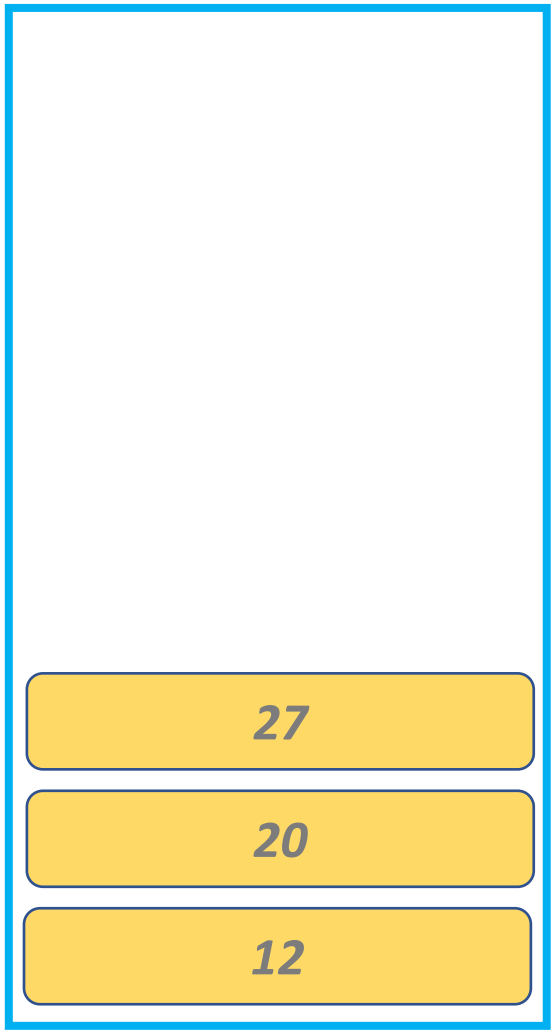


STACK

Binary Search Trees

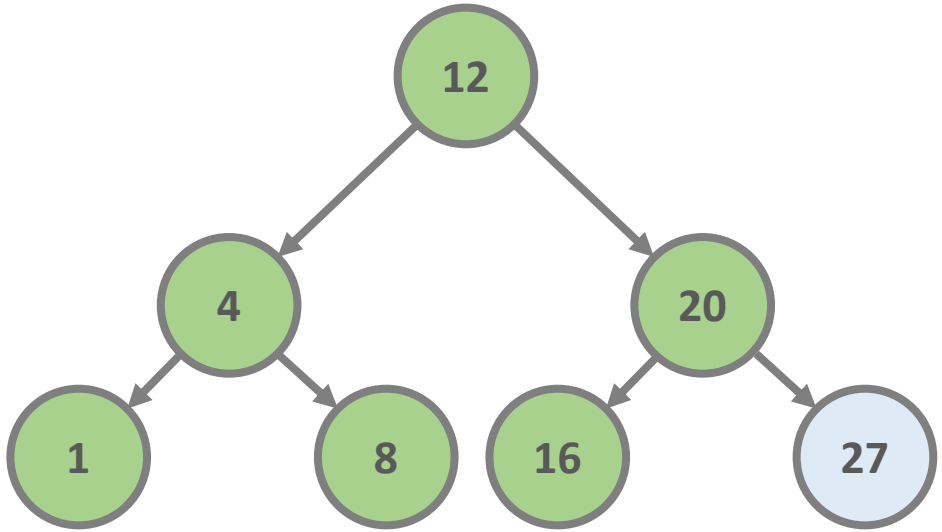



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

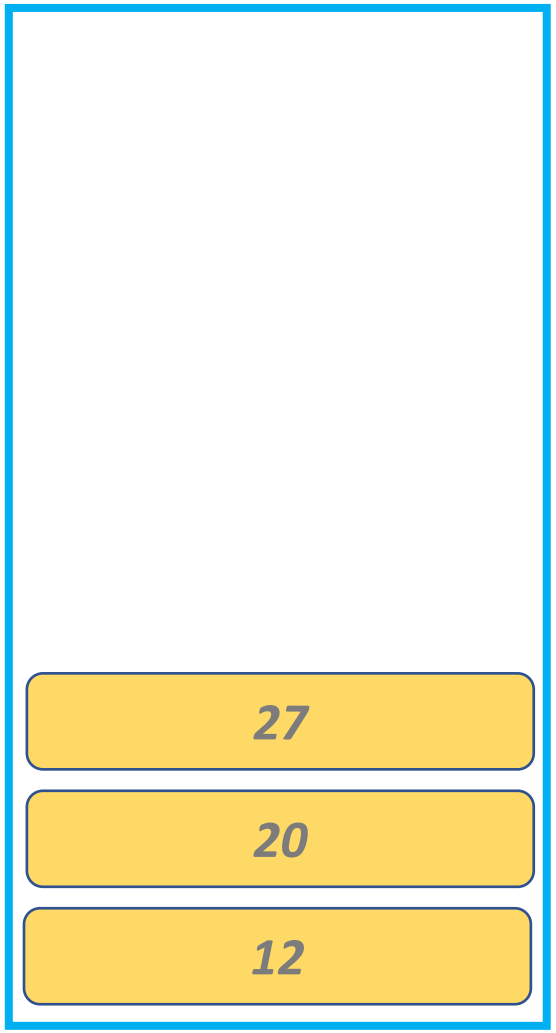


traverse(node):

if node left child is not NULL:
traverse(node left child)

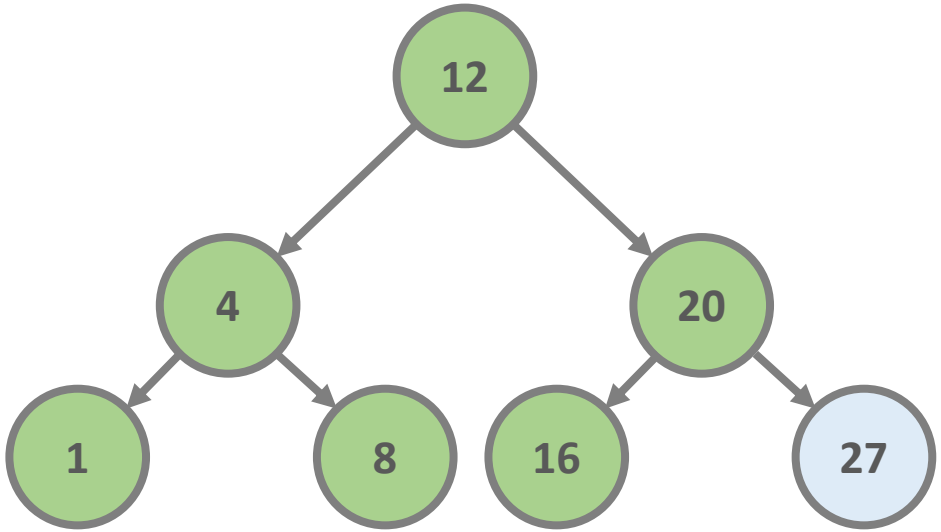
print node.data

if node right child is not NULL:
traverse(node right child)

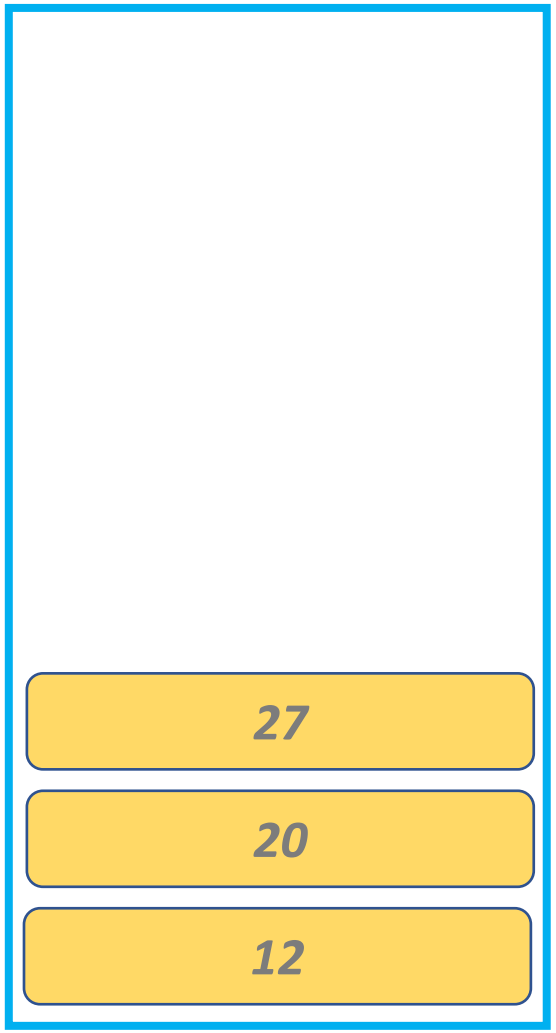


STACK

Binary Search Trees

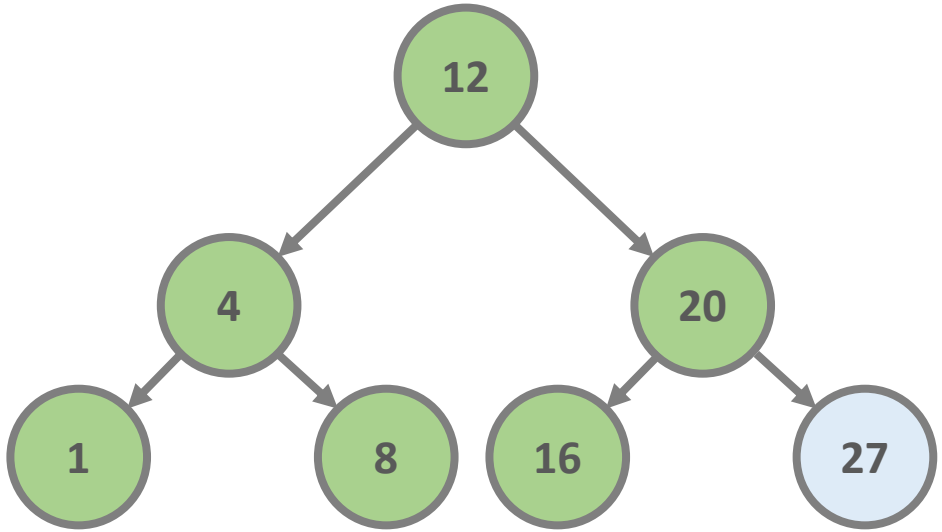



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK

Binary Search Trees

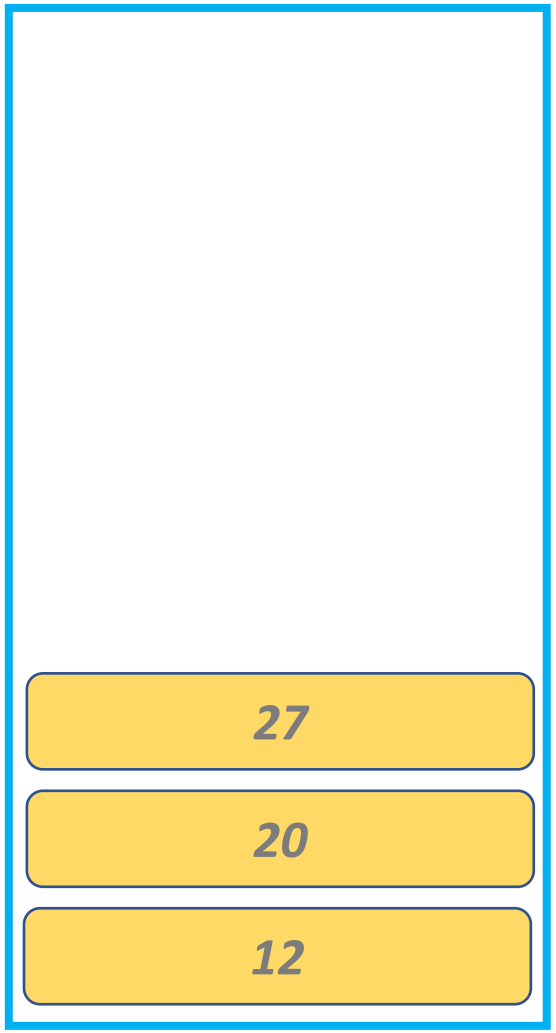


traverse(node):

if node left child is not NULL:
traverse(node left child)

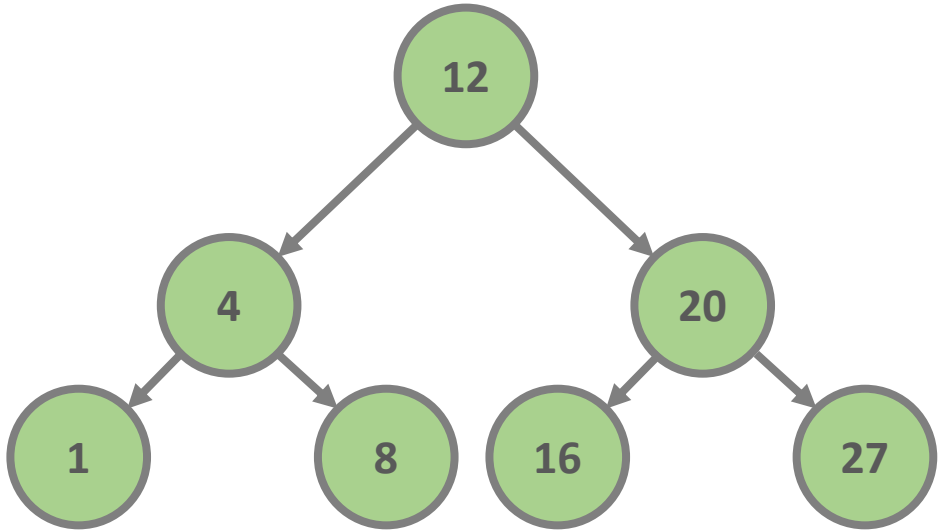
print node.data

if node right child is not NULL:
traverse(node right child)

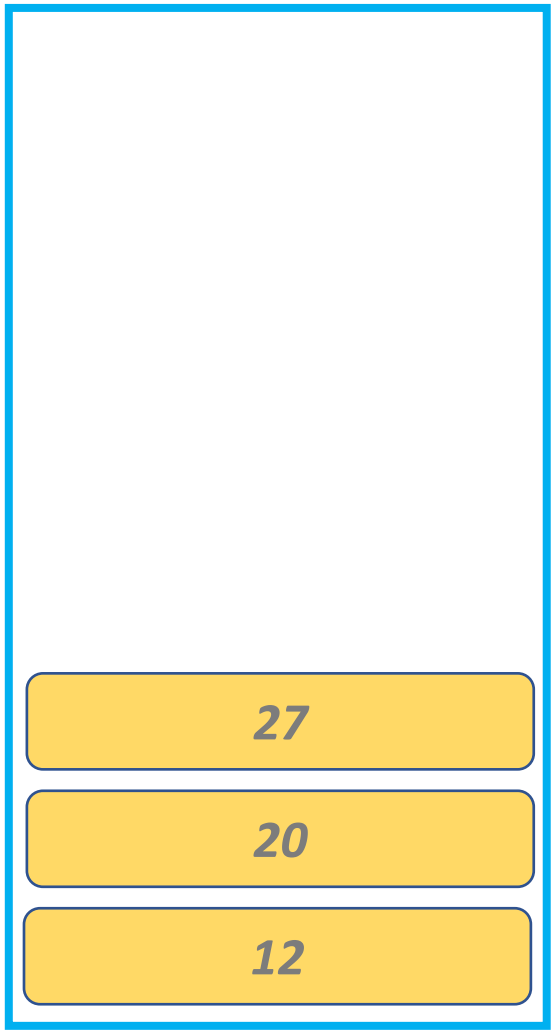


STACK

Binary Search Trees

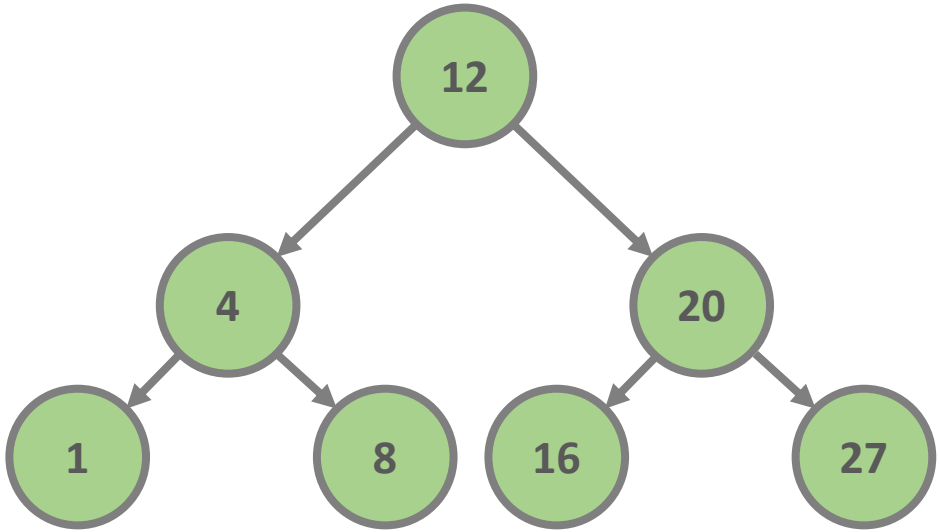



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

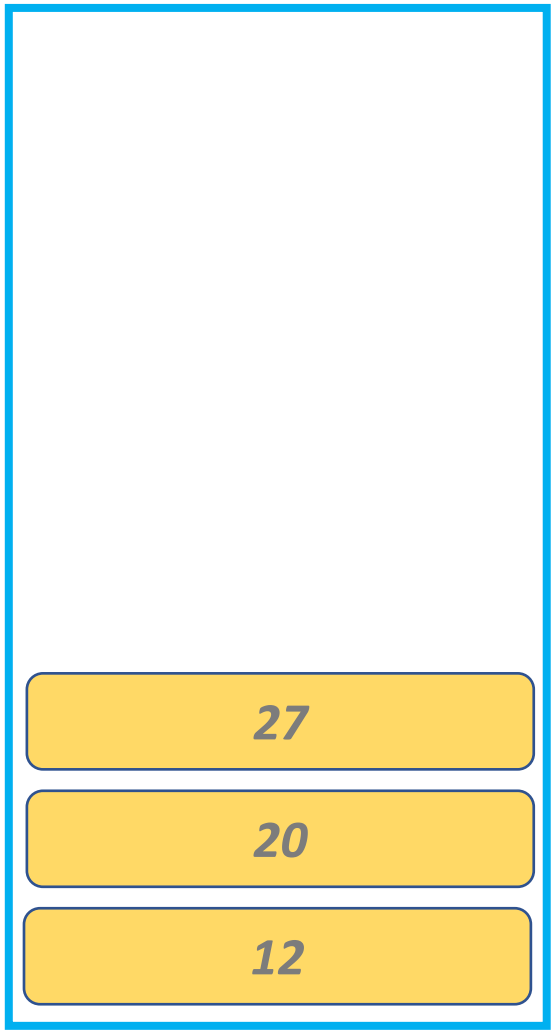


STACK

Binary Search Trees

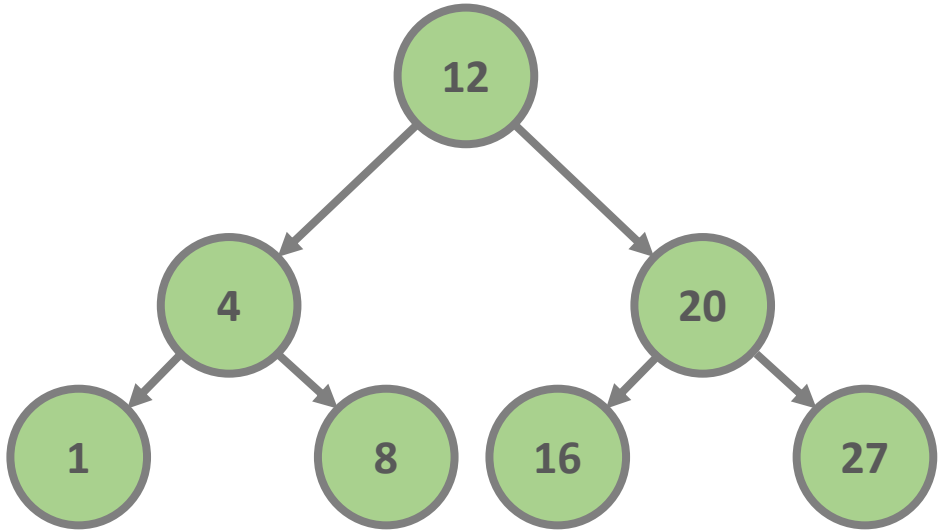



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

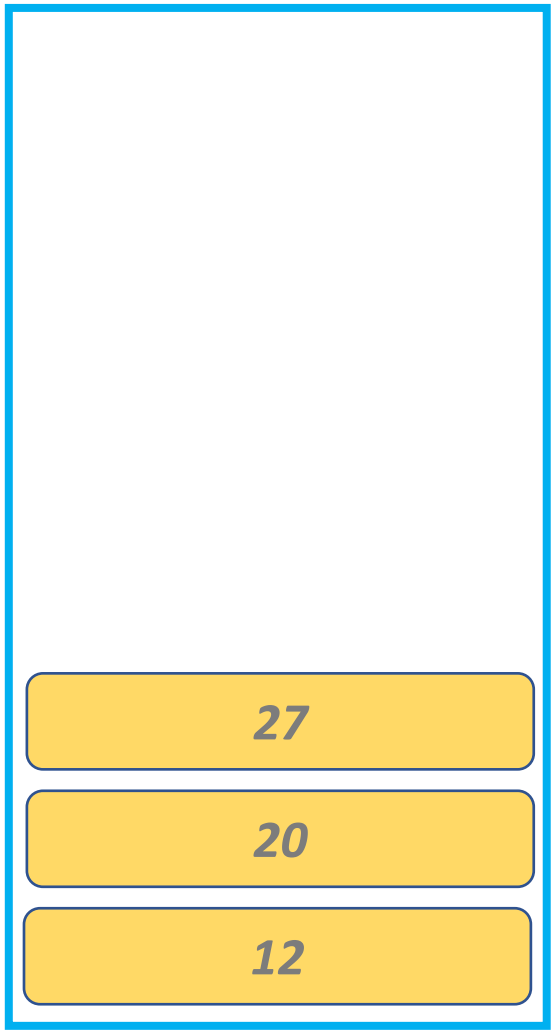


STACK

Binary Search Trees

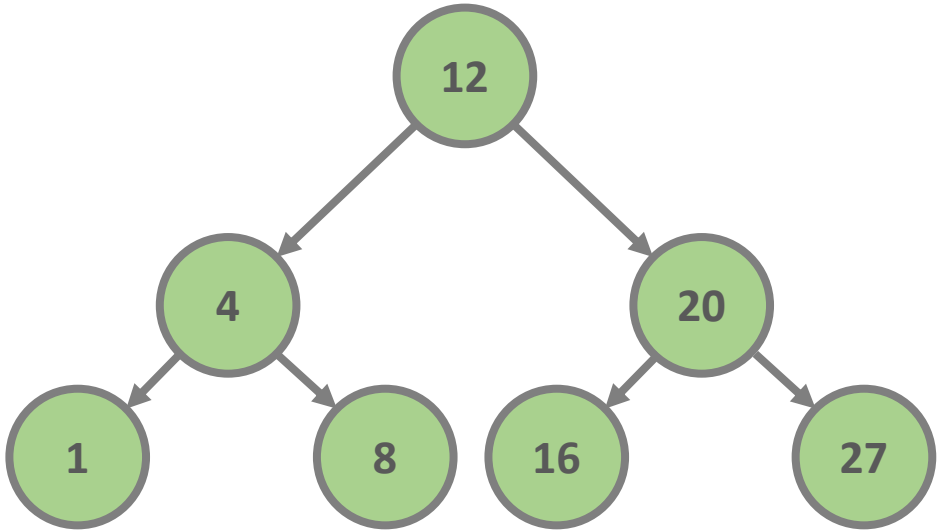



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

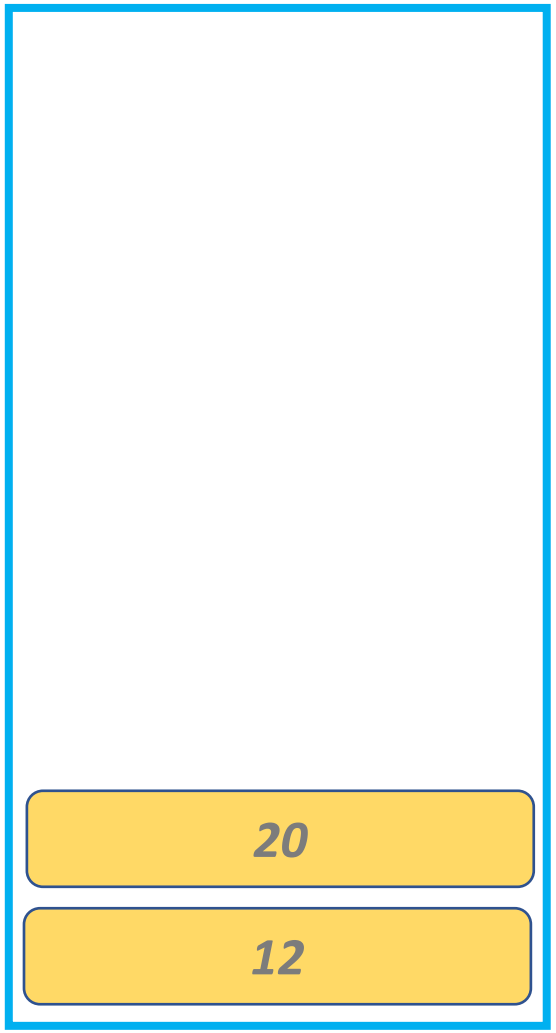


STACK

Binary Search Trees

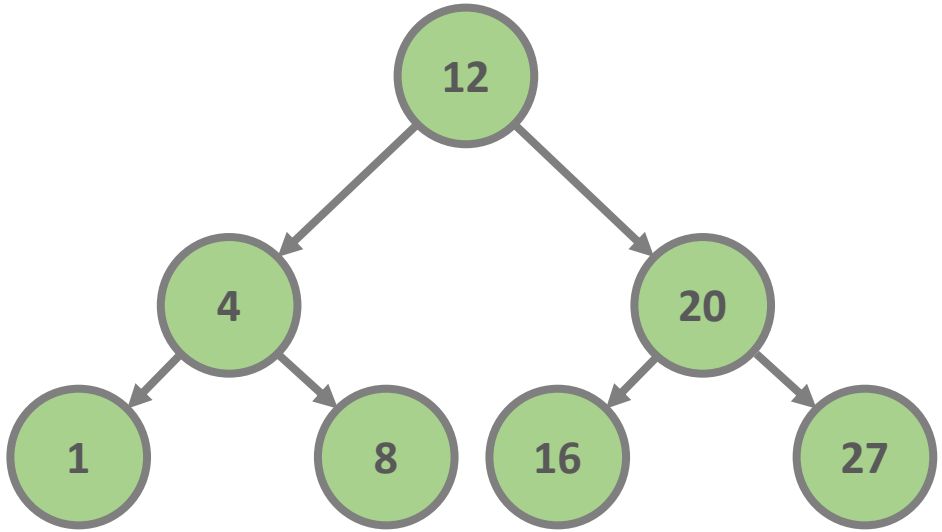



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

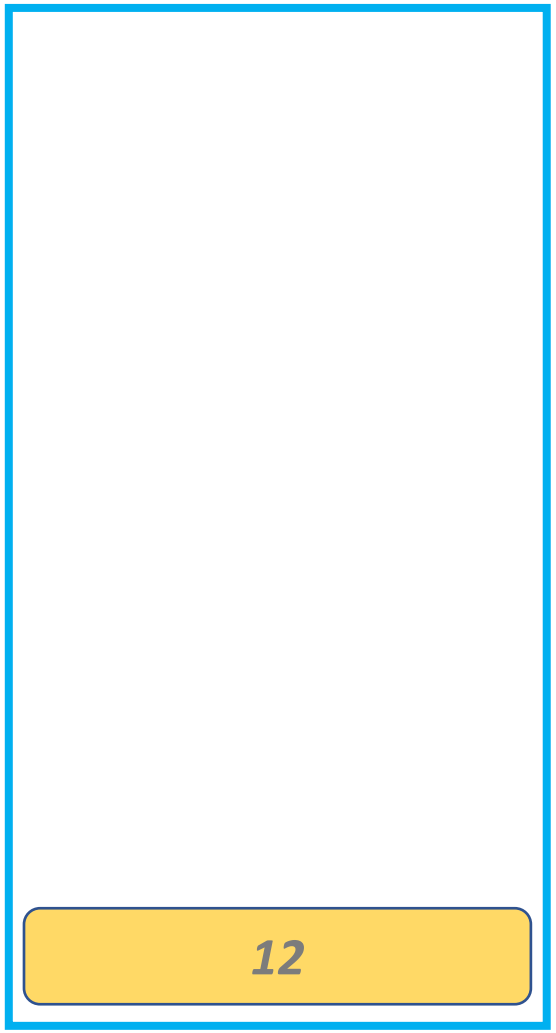


STACK

Binary Search Trees

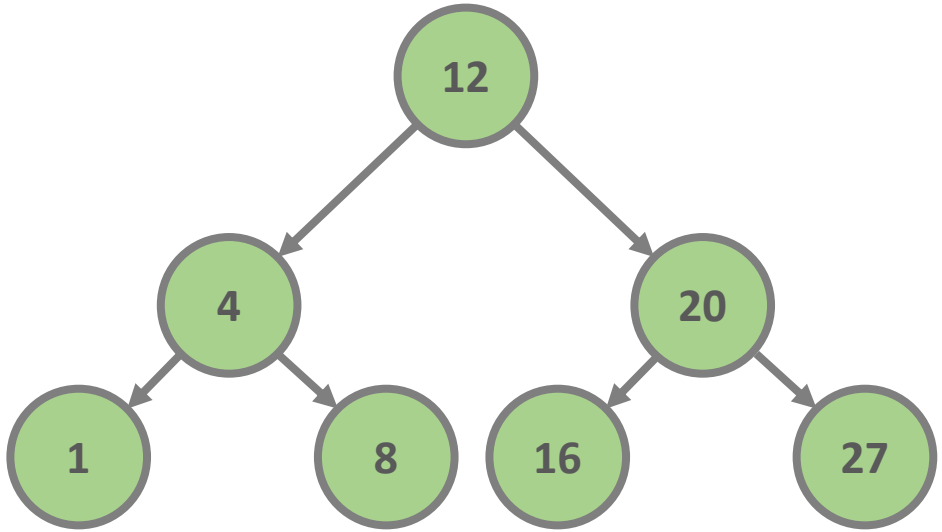



```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```

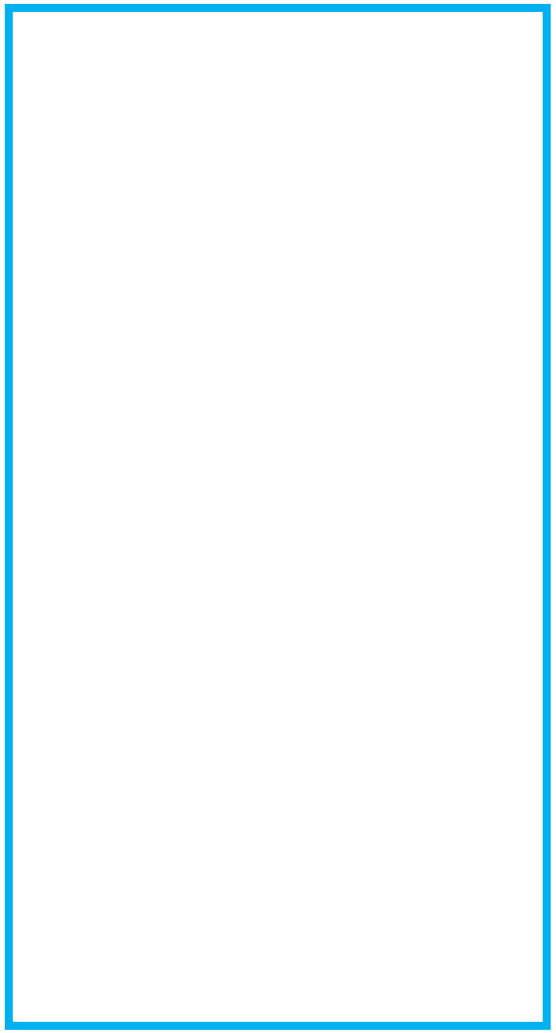


STACK

Binary Search Trees




```
traverse(node):

    if node left child is not NULL:
        traverse(node left child)

    print node.data

    if node right child is not NULL:
        traverse(node right child)
```



STACK