GRAPHS: SEARCH

David Kauchak
CS 62 – Spring 2021

1

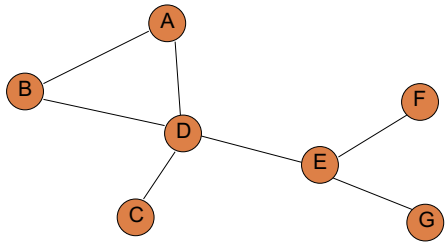
Admin

Assignment 9: Text Generation

2

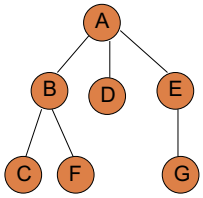
Graphs

A graph is a set of vertices V and a set of edges (u,v) ∈ E where u,v ∈ V



3

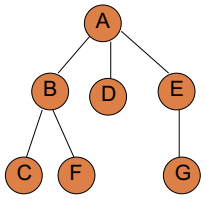
Searching a tree



How can we print out all of the vertices in a tree?

4

Searching a tree



We could do any of the traversals we saw before:
pre-order, in-order, post-order

5

A flash from the past

```
public interface Linear<E> {
    public void add(E item);
    public E remove();
    public E peek();
    public boolean empty();
}
```

Stack:
- LIFO
- Add to the back
- Remove from the back

Queue:
- FIFO
- Add to the back
- Remove from the front

6

Searching a tree

```
treeBFS( start )
    q = new Queue()
    q.add(start)
    treeSearch(q)
```

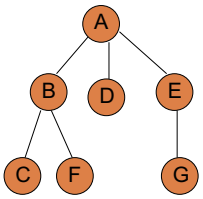
```
treeDFS( start )
    s = new Stack()
    s.add(start)
    treeSearch(s)
```

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```

7

Tree BFS

```
treeBFS( start )
    q = new Queue()
    q.add(start)
    treeSearch(q)
```

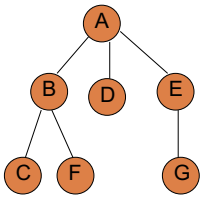


q:
Visited:

8

Tree BFS

```
treeBFS( start )
    q = new Queue()
    q.add(start)
    treeSearch(q)
```



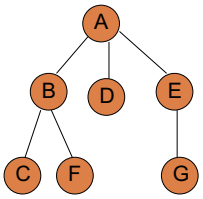
q: A

Visited:

9

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-queue: A

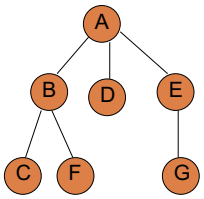
printed:

What order will the nodes get printed out?
Assume children are traversed left to right.

10

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



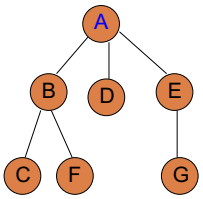
toVisit-queue: A

printed:

11

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



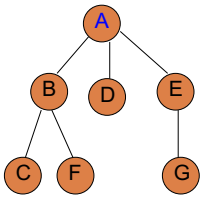
toVisit-queue:

printed: A

12

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



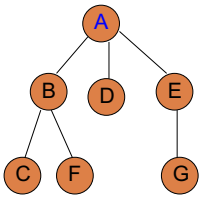
toVisit-queue:

printed: A

13

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



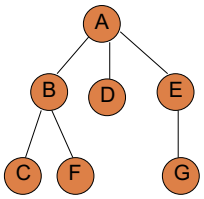
toVisit-queue: B D E

printed: A

14

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



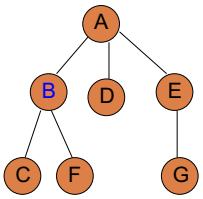
toVisit-queue: B D E

printed: A

15

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



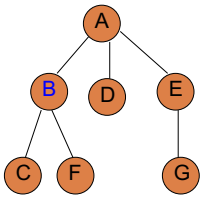
toVisit-queue: D E

printed: A B

16

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



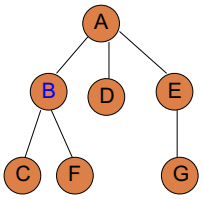
toVisit-queue: D E

printed: A B

17

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



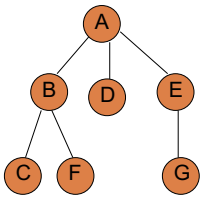
toVisit-queue: D E C F

printed: A B

18

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



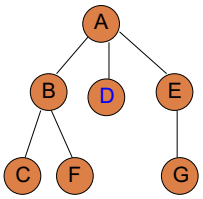
toVisit-queue: D E C F

printed: A B

19

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



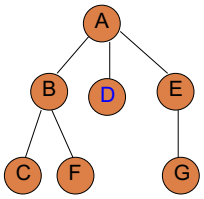
toVisit-queue: E C F

printed: A B D

20

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



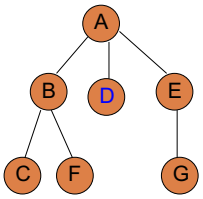
toVisit-queue: E C F

printed: A B D

21

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



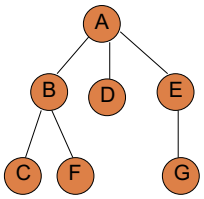
toVisit-queue: E C F

printed: A B D

22

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



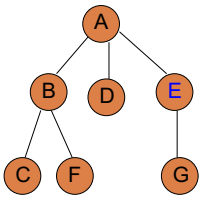
toVisit-queue: E C F

printed: A B D

23

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



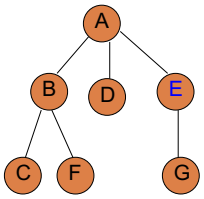
toVisit-queue: C F

printed: A B D E

24

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



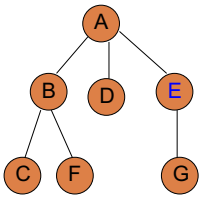
toVisit-queue: C F

printed: A B D E

25

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



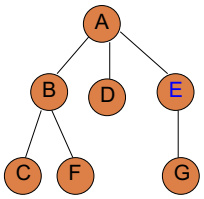
toVisit-queue: C F G

printed: A B D E

26

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-queue: C F G

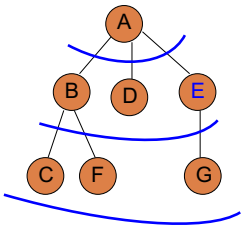
printed: A B D E

How are we exploring the vertices?

27

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-queue: C F G

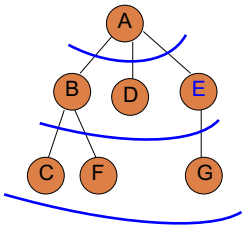
printed: A B D E

Frontier: all vertices a given number of edges from the start/root

28

Tree BFS = Tree breadth first search

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-queue: C F G

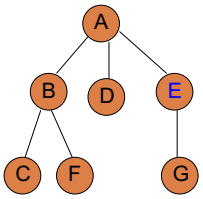
printed: A B D E

Frontier: all vertices a given number of edges from the start/root

29

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



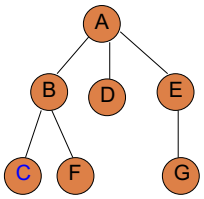
toVisit-queue: C F G

printed: A B D E

30

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



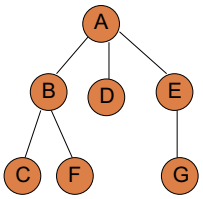
toVisit-queue: F G

printed: A B D E C

31

Tree BFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



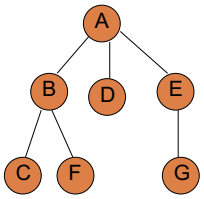
toVisit-queue:

printed: A B D E C F G

32

Tree DFS

```
treeDFS( start )
    s = new Stack()
    s.add(start)
    treeSearch(s)
```



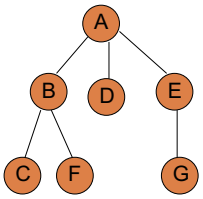
s:

printed:

33

Tree DFS

```
treeDFS( start )
    s = new Stack()
    s.add(start)
    treeSearch(s)
```



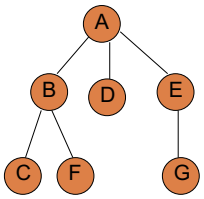
s: A

printed:

34

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-stack: A

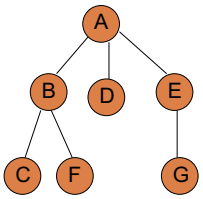
printed:

What order will the nodes get printed out?
Assume children are traversed left to right.

35

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



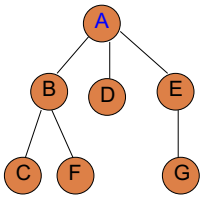
toVisit-stack: A

printed:

36

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



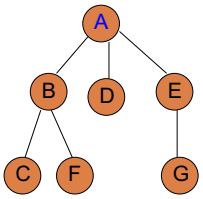
toVisit-stack:

printed: A

37

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



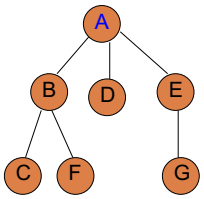
toVisit-stack:

printed: A

38

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



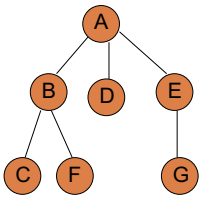
toVisit-stack: B D E

printed: A

39

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



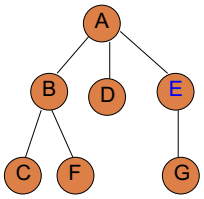
toVisit-stack: B D E

printed: A

40

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



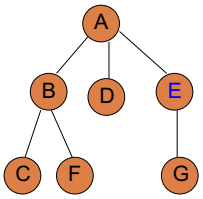
toVisit-stack: B D

printed: A E

41

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



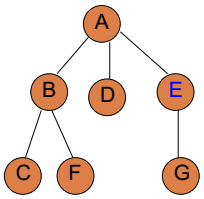
toVisit-stack: B D

printed: A E

42

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



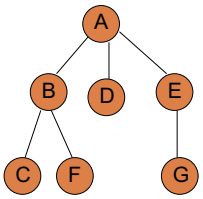
toVisit-stack: B D G

printed: A E

43

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



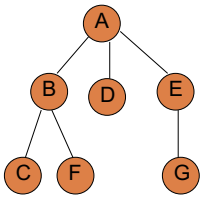
toVisit-stack: B D G

printed: A E

44

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



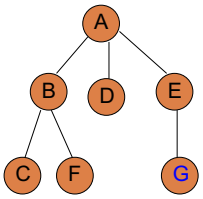
toVisit-stack: B D

printed: A E G

45

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



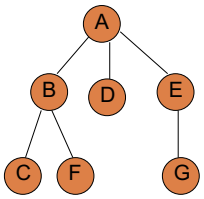
toVisit-stack: B D

printed: A E G

46

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



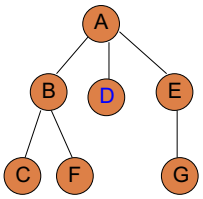
toVisit-stack: B D

printed: A E G

47

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



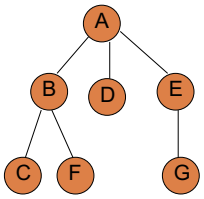
toVisit-stack: B

printed: A E G D

48

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-stack: B

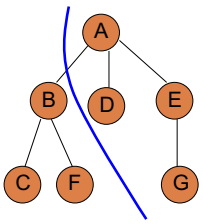
printed: A E G D

How are we exploring the vertices?

49

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-stack: B

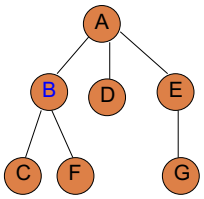
printed: A E G D

Frontier: go as far down one branch as possible, working right to left

50

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



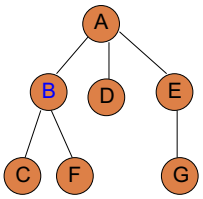
toVisit-stack:

printed: A E G D B

51

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



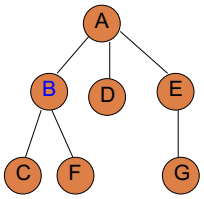
toVisit-stack:

printed: A E G D B

52

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



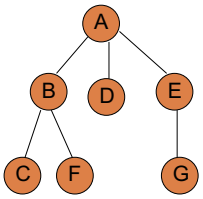
toVisit-stack: C F

printed: A E G D B

53

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



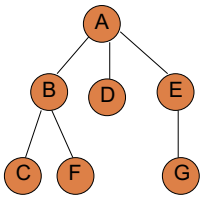
toVisit-stack: C F

printed: A E G D B

54

Tree DFS

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



toVisit-stack:

printed: A E G D B F C

55

Run-time of graph algorithms

A graph is a set of vertices V and a set of edges (u,v) ∈ E where u,v ∈ V

When we analyze graph algorithms, the run-time often includes both the number of vertices **AND** the number of edges:

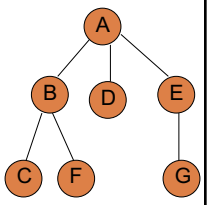
- |V| = number of vertices
- |E| = number of edges

Sometimes, in big-O notation, we'll use just V and E to represent these to simplify notation

56

treeSearch run-time

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



What is the big-O run-time of treeSearch?

Assume all of the stack/queue operations are constant.

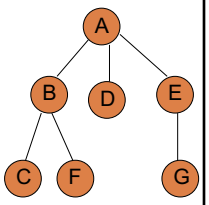
How many times do we visit each vertex?

How many times do we traverse each edge (vis the for loop)?

57

treeSearch run-time

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



How many times do we visit each vertex? Exactly once

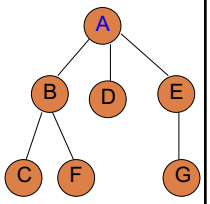
How many times do we traverse each edge (via the for loop)? Exactly once

What is the big-O run-time of treeSearch? O(|V| + |E|). Linear algorithm.

58

What algorithm is this?

```
search( v )
    // visit v, e.g., print it out
    for c in v.getChildren()
        search(c)
```

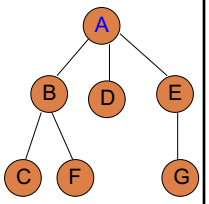


What order will the vertices get printed out?
Assume children are traversed left to right.

59

What algorithm is this?

```
search( v )
    // visit v, e.g., print it out
    for c in v.getChildren()
        search(c)
```

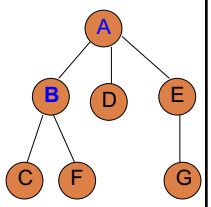


Visited: A

60

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

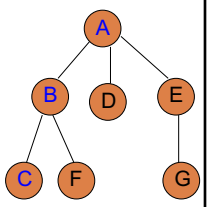


Visited: A B

61

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

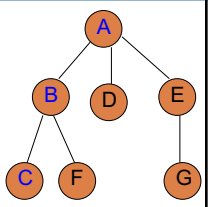


Visited: A B C

62

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

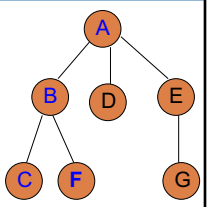


Visited: A B C Now where?

63

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

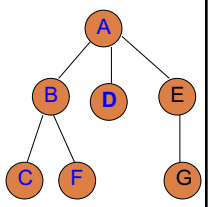


Visited: A B C F

64

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

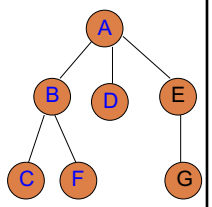


Visited: A B C F D

65

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



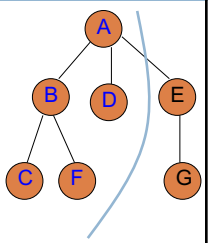
What algorithm is this?

Visited: A B C F D

66

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



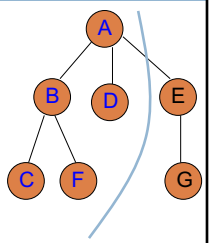
Depth first search!

Visited: A B C F D

67

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



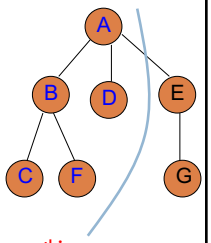
Any difference between this version and the stack version?

Visited: A B C F D

68

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



Any difference between this version and the stack version?

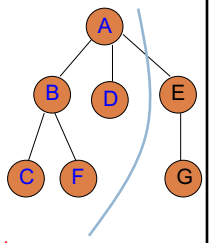
Traverses in the other direction (left to right in this case).

Visited: A B C F D

69

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



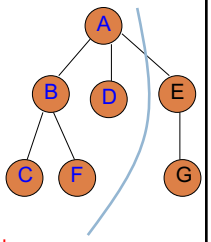
treeDFS used a stack.
Is there a stack here?

Visited: A B C F D

70

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```



treeDFS used a stack.
Is there a stack here?

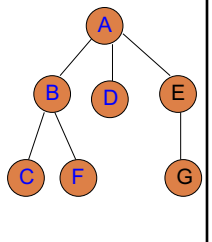
The run-time stack keeping track of recursive calls!

Visited: A B C F D

71

What algorithm is this?

```
search( v )
   // visit v, e.g., print it out
   for c in v.getChildren()
      search(c)
```

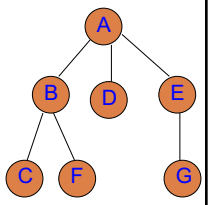


Visited: A B C F D

72

What algorithm is this?

```
search( v )
    // visit v, e.g., print it out
    for c in v.getChildren()
        search(c)
```



Visited: A B C F D E G

73

DFS versions

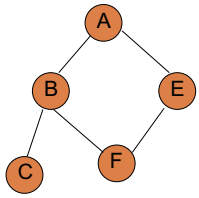
```
treeDFS( start )
    s = new Stack()
    s.add(start)
    treeSearch(s)

treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```

```
treeRecursiveDFS( v )
    // visit v, e.g., print it out
    for c in v.getChildren()
        treeRecursiveDFS(c)
```

74

treeSearch on graphs

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



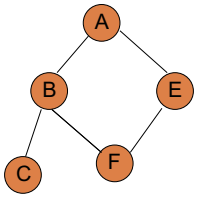
What would happen if we ran treeSearch on this graph?

Won't end!

75

treeSearch on graphs

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```



How can we fix this?

Keep track of the vertices that we've visited

76

Searching on graphs

```
treeSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        // visit v, e.g., print it out
        for c in v.getChildren()
            toVisit.add(c)
```

```
graphSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        if !visited[v]
            visited[v] = true
            for c in v.getAdjacent()
                if !visited[c]
                    toVisit.add(c)
```

77

Searching on graphs

```
graphBFS( start )
    q = new Queue()
    q.add(start)
    treeSearch(q)
```

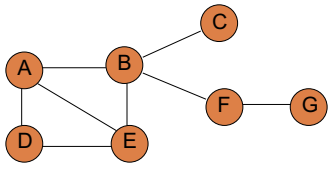
```
graphDFS( start )
    s = new Stack()
    s.add(start)
    treeSearch(s)
```

```
graphSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        if !visited[v]
            visited[v] = true
            for c in v.getAdjacent()
                if !visited[c]
                    toVisit.add(c)
```

78

BFS

```
graphBFS( start )
    q = new Queue()
    q.add(start)
    treeSearch(q)
```



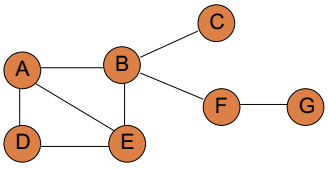
toVisit-queue: A

visited:

79

BFS

```
graphSearch( toVisit )
    while !toVisit.empty()
        v = toVisit.remove()
        if !visited[v]
            visited[v] = true
            for c in v.getAdjacent()
                if !visited[c]
                    toVisit.add(c)
```



toVisit-queue: A

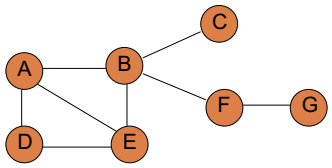
visited:

What order will the nodes get printed out?
Assume edges are traversed alphabetically.

80

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



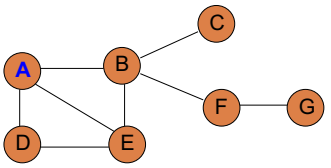
toVisit-queue: A

visited:

81

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



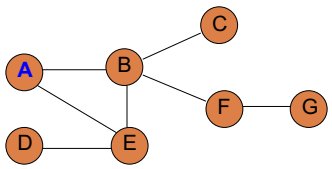
toVisit-queue:

visited: A

82

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



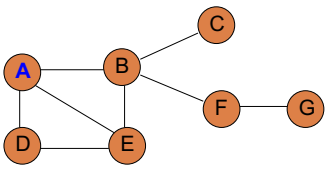
toVisit-queue:

visited: A

83

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



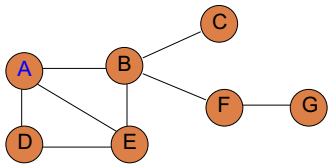
toVisit-queue: B D E

visited: A

84

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



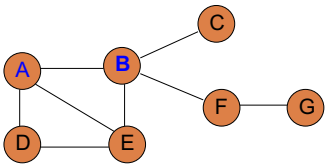
toVisit-queue: B D E

visited: A

85

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



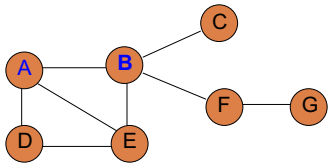
toVisit-queue: D E

visited: A B

86

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



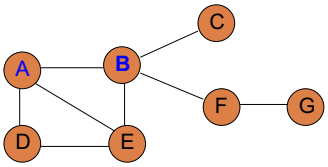
toVisit-queue: D E

visited: A B

87

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: D E C E F

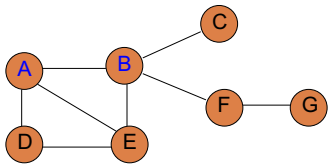
visited: A B

Notice that we do add E again to toVisit since we haven't visited it yet

88

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



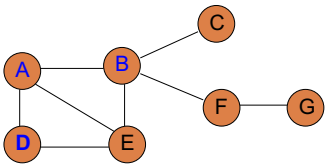
toVisit-queue: D E C E F

visited: A B

89

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



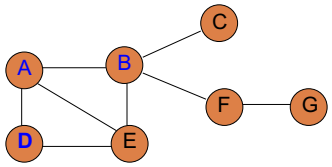
toVisit-queue: E C E F

visited: A B D

90

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



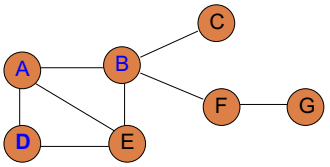
toVisit-queue: E C E F

visited: A B D

91

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: E C E F E

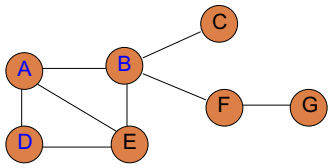
visited: A B D

We add E again to toVisit since we still haven't visited it yet

92

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



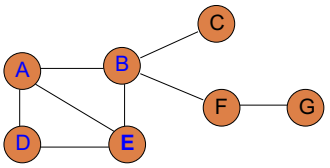
toVisit-queue: E C E F E

visited: A B D

93

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



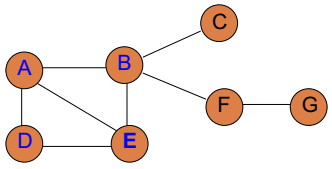
toVisit-queue: E C E F E

visited: A B D E

94

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



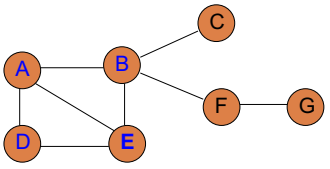
toVisit-queue: E C E F E

visited: A B D E

95

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: E C E F E

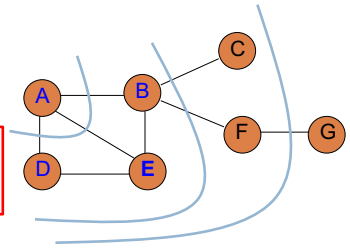
visited: A B D E

No adjacent vertices that haven't been visited

96

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: E C E F E

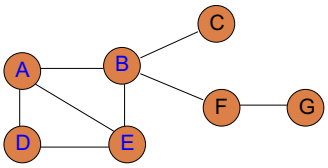
visited: A B D E

Frontier: all vertices a given number of edges from the start/root

97

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



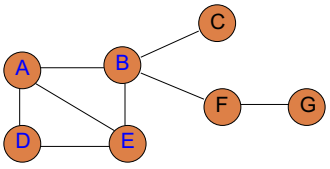
toVisit-queue: E C E F E

visited: A B D E

98

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: C E F E

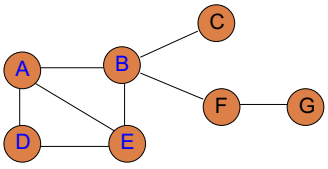
visited: A B D E

E has already been visited

99

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



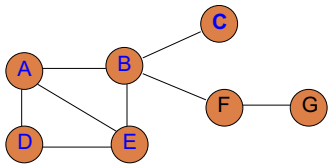
toVisit-queue: C E F E

visited: A B D E

100

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



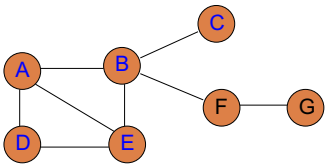
toVisit-queue: E F E

visited: A B D E C

101

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



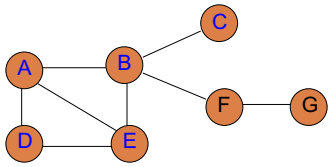
toVisit-queue: E F E

visited: A B D E C

102

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue: F E

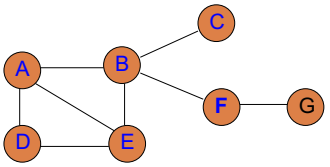
visited: A B D E C

E has already been visited

103

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



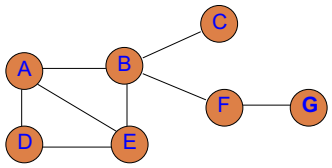
toVisit-queue: E G

visited: A B D E C F

104

BFS

```
graphSearch( toVisit )
  while !toVisit.empty()
    v = toVisit.remove()
    if !visited[v]
      visited[v] = true
      for c in v.getAdjacent()
        if !visited[c]
          toVisit.add(c)
```



toVisit-queue:

visited: A B D E C F G

105