

Python IV - Lesson 26

Date: Dec 26, 2022

Agenda

- ▶ CCC questions
- ▶ BFS



Proverbs 11:17

- ▶ “Those who are kind benefit themselves, but the cruel bring ruin on themselves.”

Recursion

Recursion is the process of defining something in terms of itself.

We know that a function can call other functions. It is even possible for the function to call itself.

Most important things:

1. Find the base case.
2. Find the sub-problem.

CCC questions

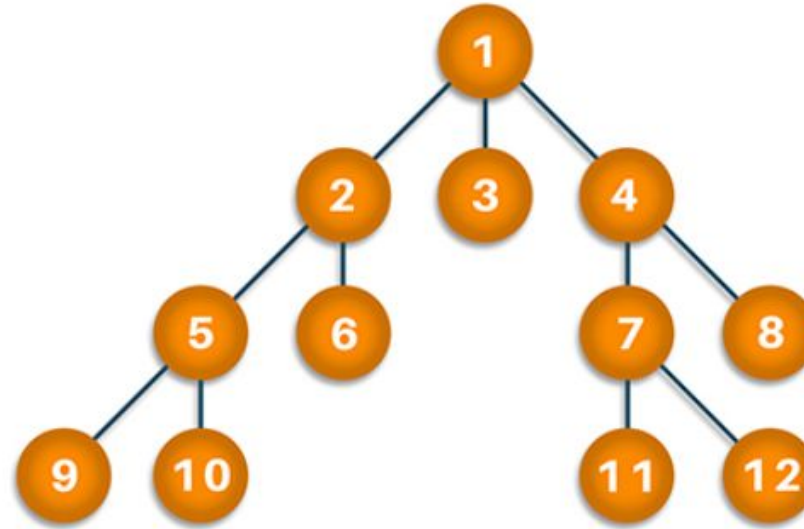
Introduction:

<https://cemc.uwaterloo.ca/contests/computing/details.html>

Past contests:

https://www.cemc.uwaterloo.ca/contests/past_contests.html

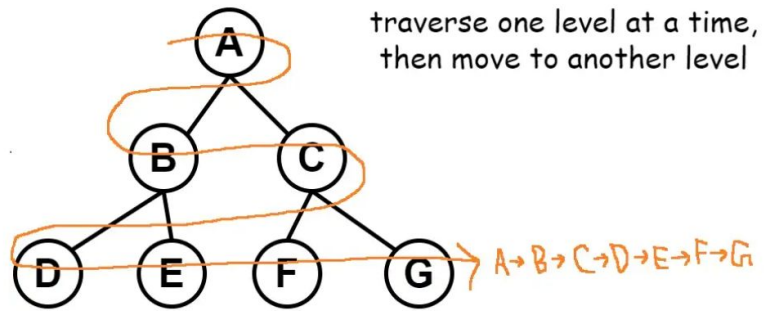
Breadth-First Search (BFS)



BREADTH FIRST SEARCH

Breadth-First Search (BFS) is an algorithm used for traversing graphs or trees. Traversing means visiting each node of the graph. Breadth-First Search is a recursive algorithm to search all the vertices of a graph or a tree. BFS in python can be implemented by using data structures like a dictionary and lists.

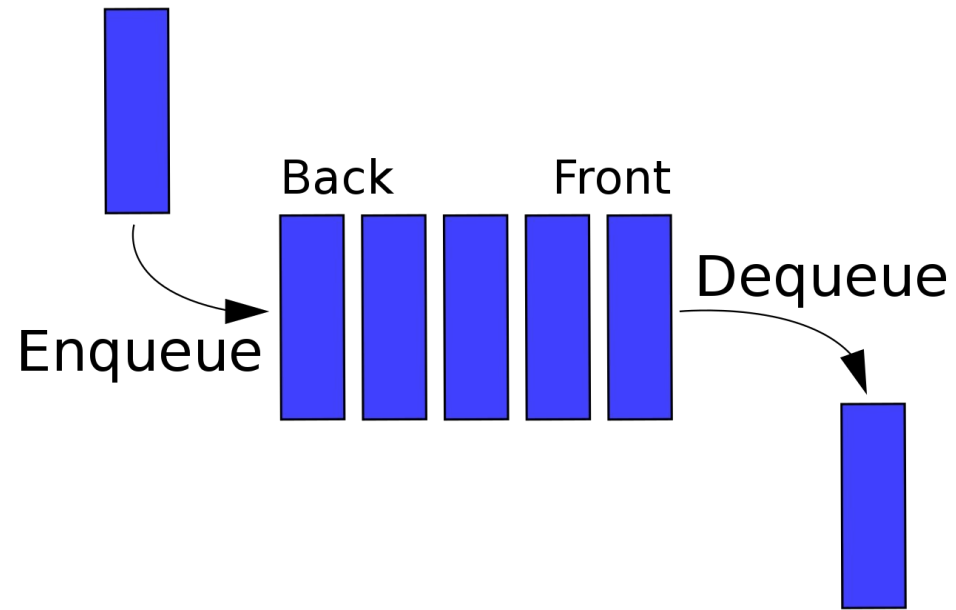
Breadth-First Search (BFS)



This traversal technique is easy to understand for us humans. But we need to think about how a computer can understand it.

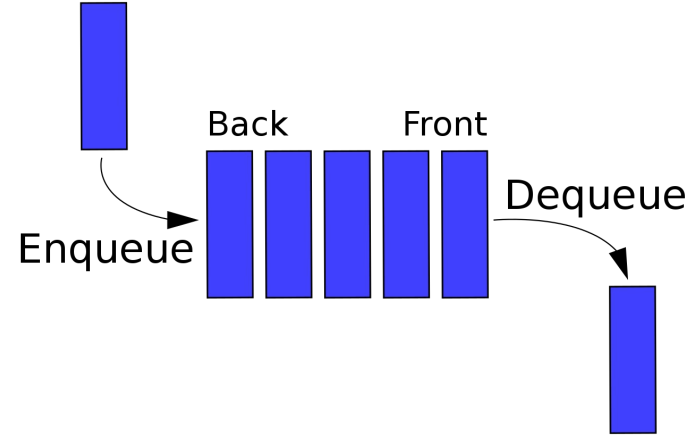
The **Queue** data structure can help us with this.

Queue



Queue is a linear structure of data that follows the First-In-First-Out (**FIFO**) principle. The element entering first in a queue will also leave first.

Queue



There are various ways to implement a queue in Python.

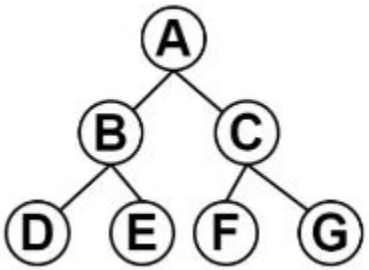
This lecture we will implement queue using **list**.

List is a Python's built-in data structure that can be used as a queue. Instead of `enqueue()` and `dequeue()`, **`append()`** and **`pop()`** function is used.

Try to implement the queue using list in Colab!


Queue

The FIFO principle of a queue will ensure the level-wise searching of BFS. Let's see how from the sketch below:



Queue: push A

(A)

Queue: pop A and push its child nodes B & C

 BFS: A

Queue: pop B and push its child nodes D & E
 BFS: A, B

pop C and push its child nodes F & G

Queue: G F E D → C

BFS: A, B, C

pop D, it has no child so push nothing

Queue: (G) (F) (E) → (D)

BFS: A, B, C, D

pop E, it has no child so push nothing
Queue: (G) (F) → (E)
BFS: A, B, C, D, E

pop F, it has no child so push nothing
Queue: (G) → (F)
BFS: A, B, C, D, E, F

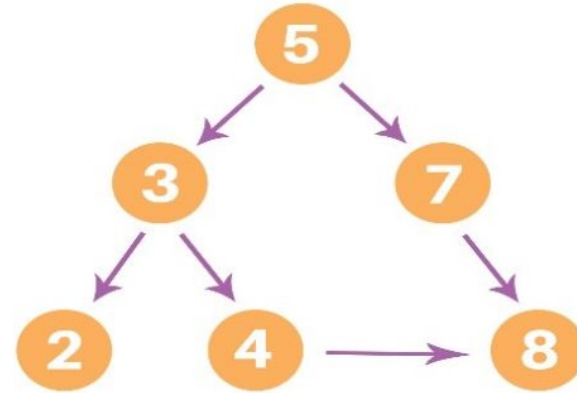
Queue: pop G, it has not child so push nothing
BFS: A, B, C, D, E, F, G

Queue is not empty, so stop traversing

Breadth-First Search (BFS)

The pseudocode for BFS in python goes as below:

1. Start with a root node and push it to the queue.
2. Mark the root node as visited and print it
3. Continue a loop until the queue is empty:
 - 3.1. Pop the front node from the queue
 - 3.2. Push the child/neighbor nodes of the front node to the queue
 - 3.3 Mark them as visited and print

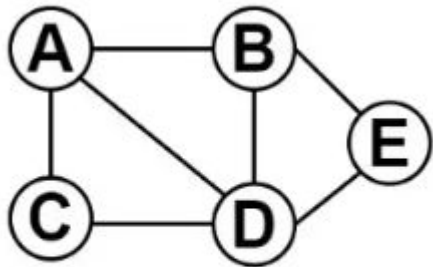


Let's write the code in Colab!

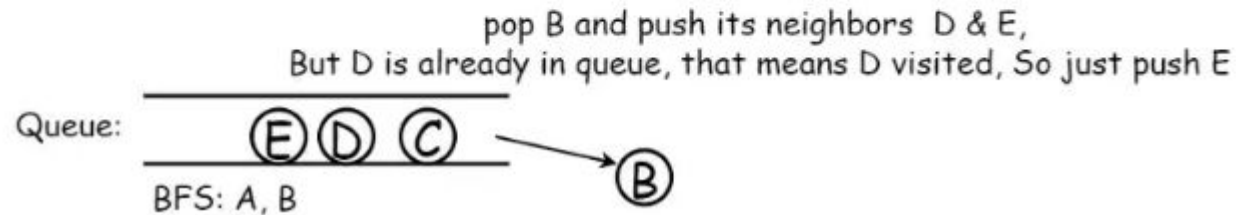
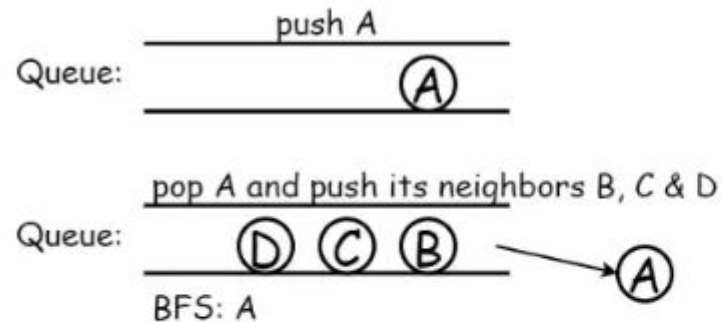
Breadth-First Search (BFS)

a little bit complex graph that contains cycles in it:

We can't determine child nodes of a node in an undirected graph that contains cycles. Rather we will use the concept of 'neighbor nodes' here. The nodes directly connected to a node are the neighbors of that node.

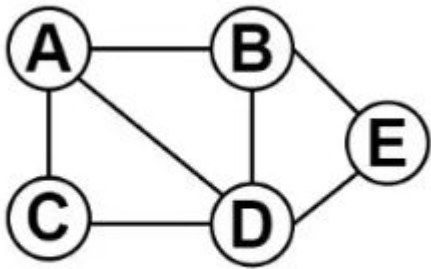


Let's Choose A as root node

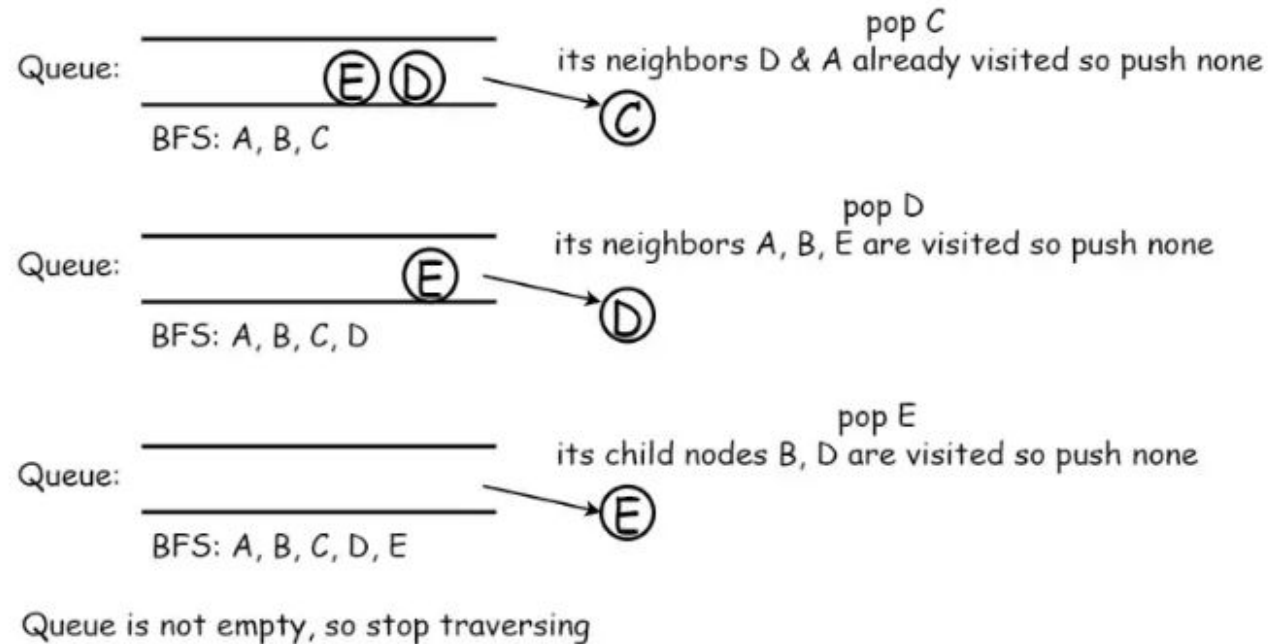


Breadth-First Search (BFS)

This graph shows us why we need to check already visited nodes. We do it to avoid traversing the same node more than once.



Let's Choose **A** as root node



BFS Exercise

<https://leetcode.com/problems/binary-tree-level-order-traversal/>

Introduction to Topological Sorting

<https://www.geeksforgeeks.org/topological-sorting/>