**CS469 Data Structures and Algorithms**

**HOS06 Queues and Breadth-First Search**

05/03/2023 Updated by Christopher Sharp

02/09/2024 Reviewed by Anh Nguyen

09/29/2024 Reviewed by Shahid Khan

School of Technology and Computing (STC) @City University of Seattle (CityU)

**Before You Start**

* The document’s examples are written in Python. If you don’t know Python programming language, please finish the Python tutorial in Module00 folder before you start the assignment.
* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

* Understand what a breadth first graph is, and what it is used for.
* Learn how to create graphs in Python.

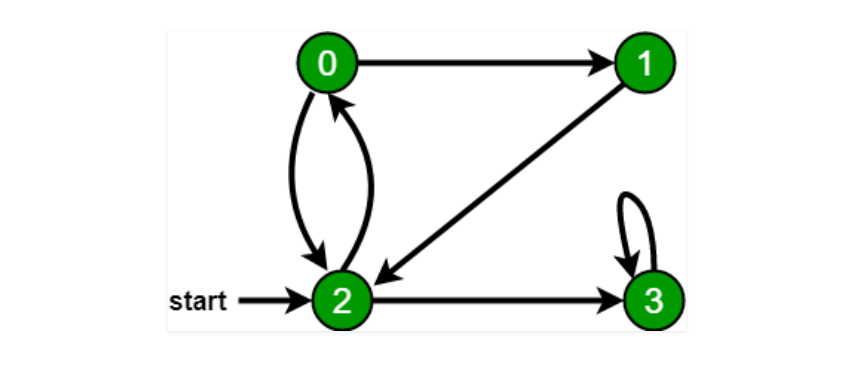
**Resources**

* *Breadth first search or BFS for a graph*. GeeksforGeeks. (2022, April 28). Retrieved May 11, 2022, from <https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/>
* *Applications of breadth first traversal*. GeeksforGeeks. (2019, December 10). Retrieved May 11, 2022, from <https://www.geeksforgeeks.org/applications-of-breadth-first-traversal/>
* *Breadth first search algorithm | shortest path* . (2019, December 29). Retrieved May 11, 2022, from <https://www.youtube.com/watch?v=oDqjPvD54Ss>
* Khan Academy. (n.d.). *The breadth-first search algorithm (BFS) (article)*. Khan Academy. Retrieved May 11, 2022, from <https://www.khanacademy.org/computing/computer-science/algorithms/breadth-first-search/a/the-breadth-first-search-algorithm>

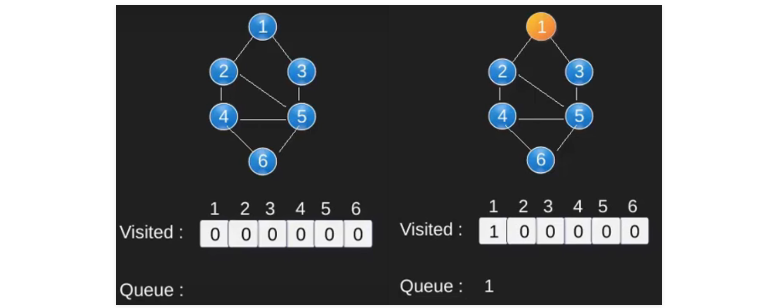
# Breadth-First Traversal

A breadth-first graph is a graph that looks for the shortest path from point A to point B. A good example of when it may be best to use a breadth-first graph is when you are trying to determine what is the fastest route to your destination.

*Challenge: When would you choose a bread-first search?*



Queues in data work like how they do in real life, unlike a stack, queues work on a first-in, first-out basis. The implementation of a breadth-first graph uses queues to track which nodes have been visited and which ones haven’t.



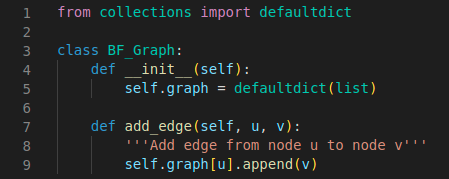
Reference to [third](https://www.khanacademy.org/computing/computer-science/algorithms/breadth-first-search/a/the-breadth-first-search-algorithm) graph on Khan Academy for an interactive example of a breadth-first graph with a queue.

*Challenge: What happens if you add more edges?*

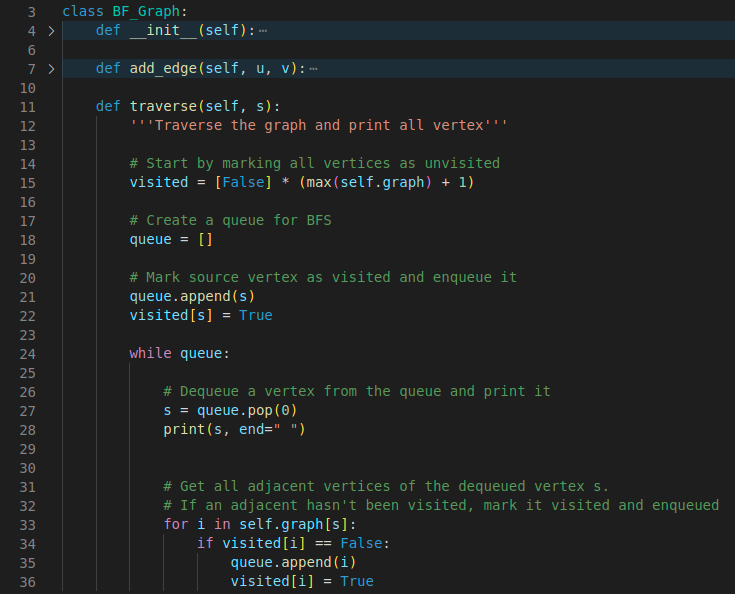
## Implementation

To make a breadth-first graph in Python, first we must start with using collections, then defining the graph. Notice that we are implementing the green node graph above.

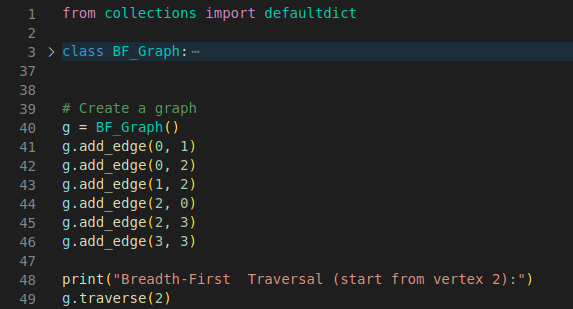
*In your Module 6 HOS folder create a file named* ***BFS.py*** *and type in the following.*



After defining the graph, you can start by making the queue, or rather, the order in which the graph will go.



From here, we can add edges to graph, each edge will map a path to another.



The output should look like this.



# Push Your Work to Github

Open terminal and make sure you’re in the repository folder. (i.e: hos06\_courseName\_GitHubUserName)

**Type the following command to upload your work**:

>>>> git add .

>>>> git commit -m “Submission for HOS06”

>>>> git push origin master