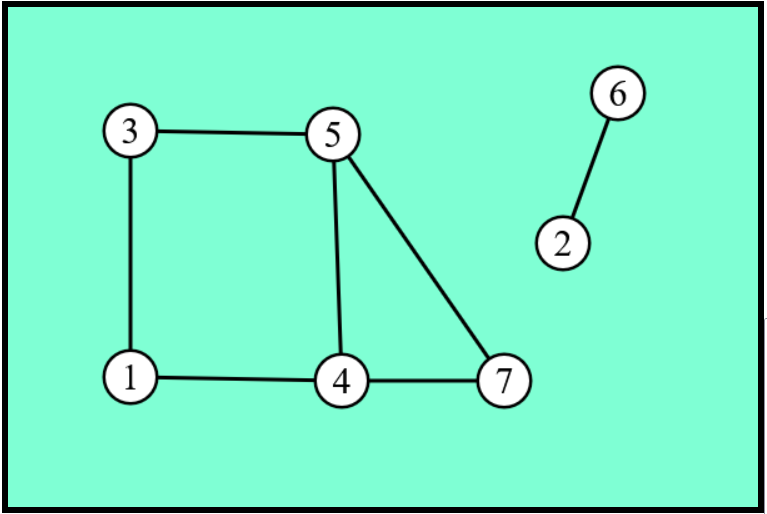
**Обход графа в ширину**

Обход в глубину может решать большинство задач, связанных с достижимостью и поиском путей в графах. Но иногда нужно найти не любой путь, а кратчайший. Например, для рисунка ниже обход в глубину находил путь 1-3-5-4-7 из 1 в 7, но кратчайший путь 1-4-7.

Эту задачу можно решать обходом в ширину (breadth-first search, BFS).



Идея обхода в ширину в том, чтобы перебирать вершины в другом порядке. Сначала рассмотрим только стартовую вершину, потом все вершины на расстоянии 1, потом все вершины на расстоянии 2 и т.д.

Например, для примера выше и стартовой вершины 1 порядок обхода будет следующим: 1 3 4 5 7 (вершины 3 и 4 находятся на расстоянии 1 от стартовой, а вершины 5 и 7 – на расстоянии 2).

**Как перебирать вершины в таком порядке:** используем очередь. Пусть сначала в очереди находится только стартовая вершина. На каждом шаге алгоритма из очереди достаётся первая вершина и перебираются все рёбра из неё. Если ребро ведёт в ещё не посещённую вершину, то она добавляется в конец очереди.

**Когда нужно отмечать вершину как посещённую?**

Чтобы вершина не добавлялась в очередь несколько раз, нужно её отмечать как посещённую сразу при первом добавлении в очередь. Стартовая вершина отмечается как посещённая в самом начале алгоритма.

Теперь можно более конкретно описать шаги алгоритма:

1. Отметить стартовую вершину как посещённую, добавить её в очередь

2. Пока очередь не пустая, повторять следующее:

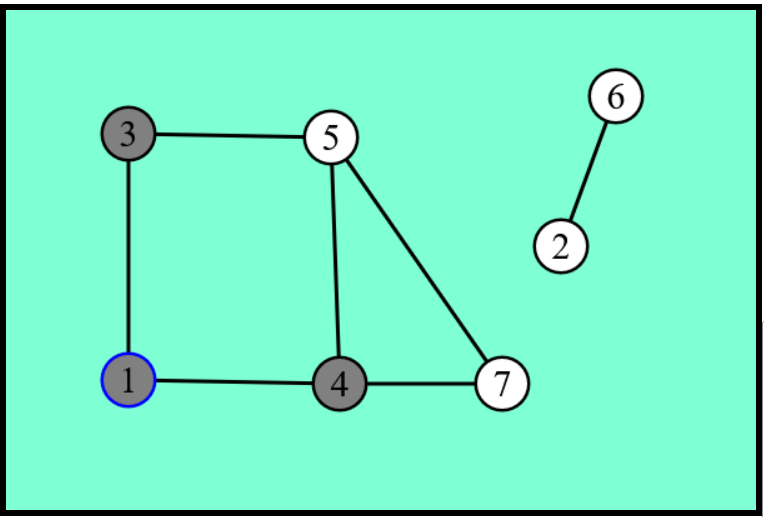
2.1. Достаём вершину из начала очереди.

2.2. Перебираем все рёбра из неё. Если ребро ведёт в непосещённую вершину, то отмечаем её как посещённую и добавляем в конец очереди.

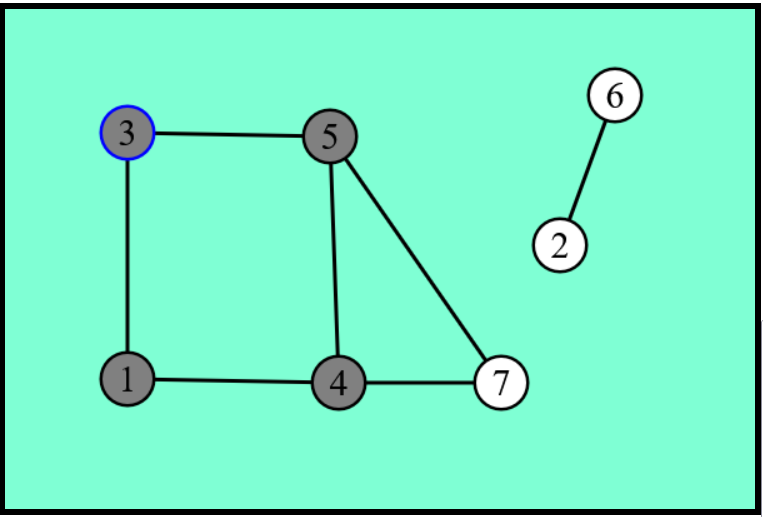
Рассмотрим работу алгоритма на примере выше:

1. Отмечаем стартовую вершину 1 как посещённую, добавляем её в очередь.

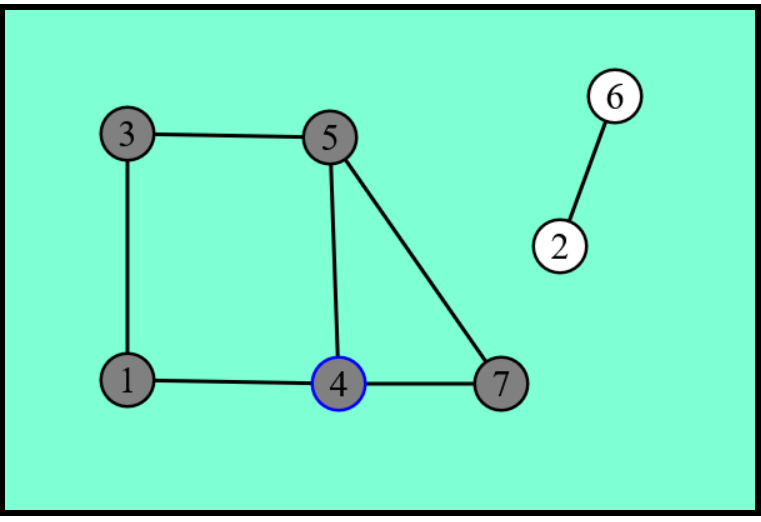
2. Достаём первую вершину очереди (1), перебираем рёбра из неё. Рёбра 1-3 и 1-4 оба ведут в новые вершины, отмечаем их как посещённые и добавляем в очередь. Сейчас очередь имеет вид [3, 4].



3. Достаём первую вершину из очереди (3), перебираем рёбра из неё. Ребро 3-1 ведёт в посещённую вершину, пропускаем. Ребро 3-5 ведёт в новую вершину, отмечаем её как посещённую и добавляем в конец очереди. Очередь имеет вид [4, 5].



4. Достаём первую вершину из очереди (4), перебираем рёбра из неё. Ребро 4-1 ведёт в посещённую вершину, пропускаем. Также пропускаем ребро 4-5. Ребро 4-7 ведёт в новую вершину, отмечаем её как посещённую и добавляем в конец очереди. Очередь имеет вид [5, 7].



5. Достаём первую вершину из очереди (5), перебираем рёбра из неё. Рёбра 4-1, 4-5, 4-7 ведут в посещённые вершины, пропускаем. Очередь имеет вид [7].

6. Достаём первую вершину из очереди (7), перебираем рёбра из неё. Рёбра 7-4 и 7-5 ведут в посещённые вершины, пропускаем. Очередь имеет вид [].

7. Очередь пуста, завершаем алгоритм.

**Как восстанавливать кратчайший путь**

Используем ту же идею, что и в обходе в глубину. Храним для каждой вершины предка – вершину, из которой мы пришли в данную. Чтобы найти кратчайший путь от стартовой вершины до конечной, пройдём из конечной вершины по предкам, пока не придём в стартовую.