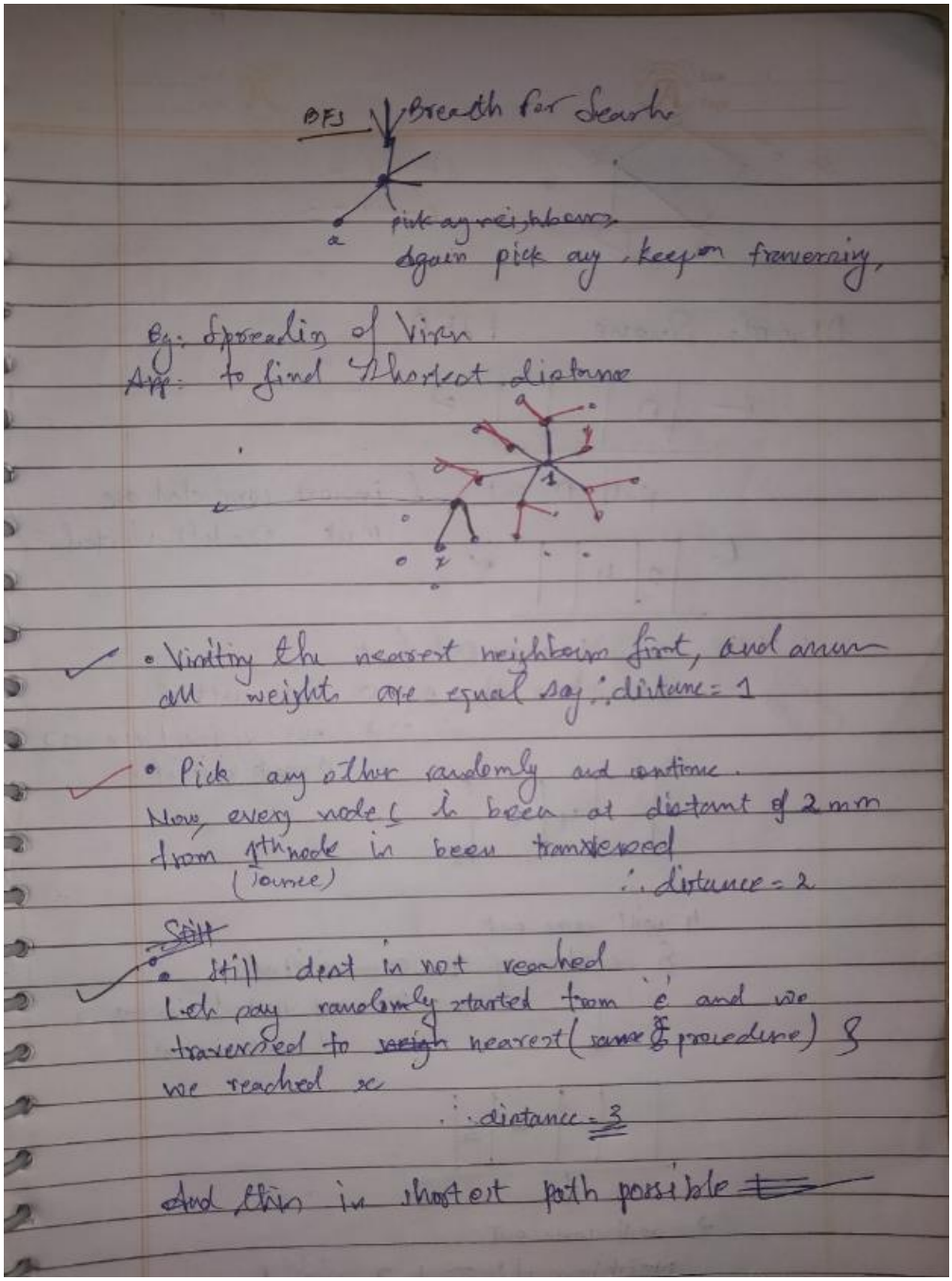
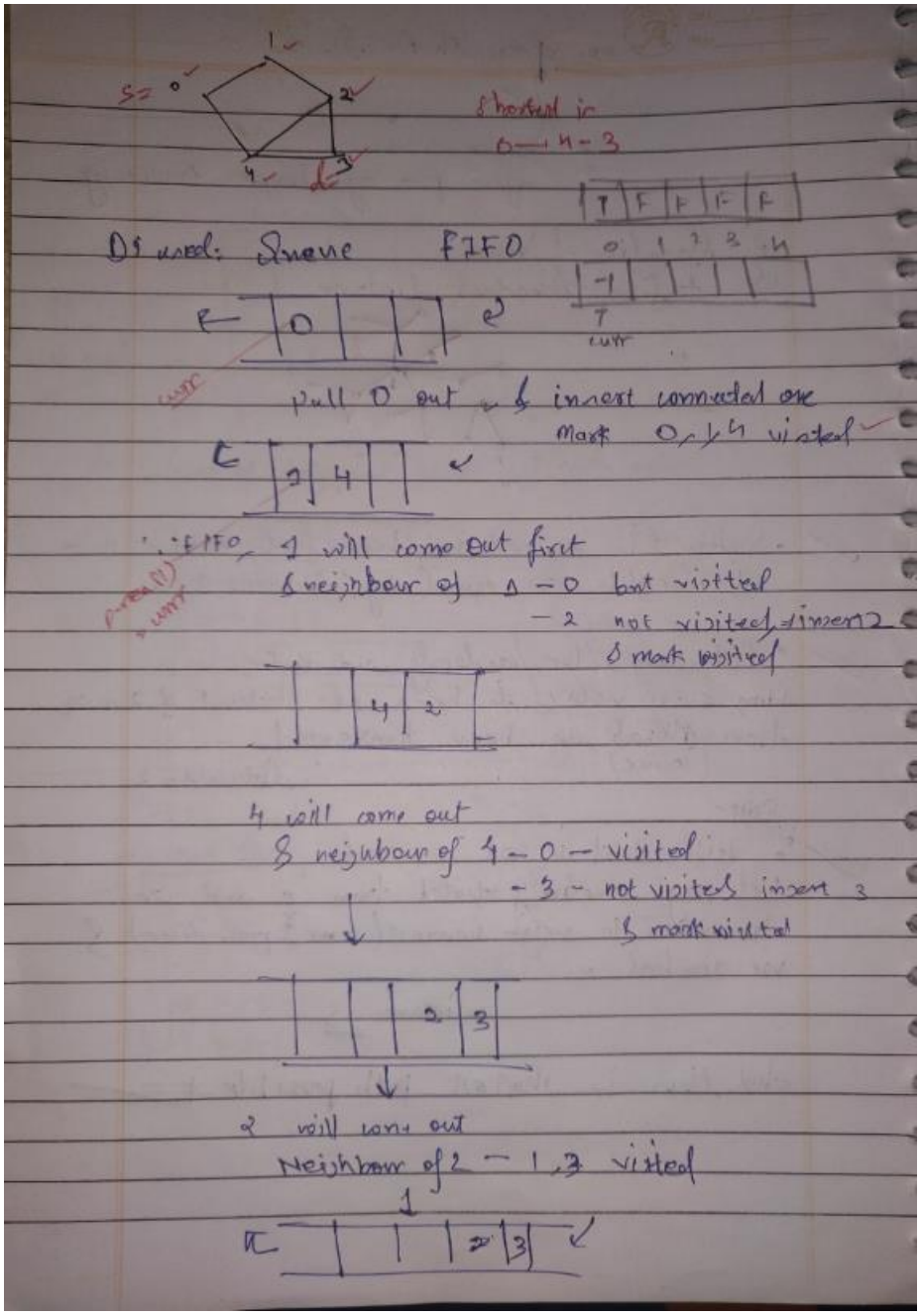
Bfs

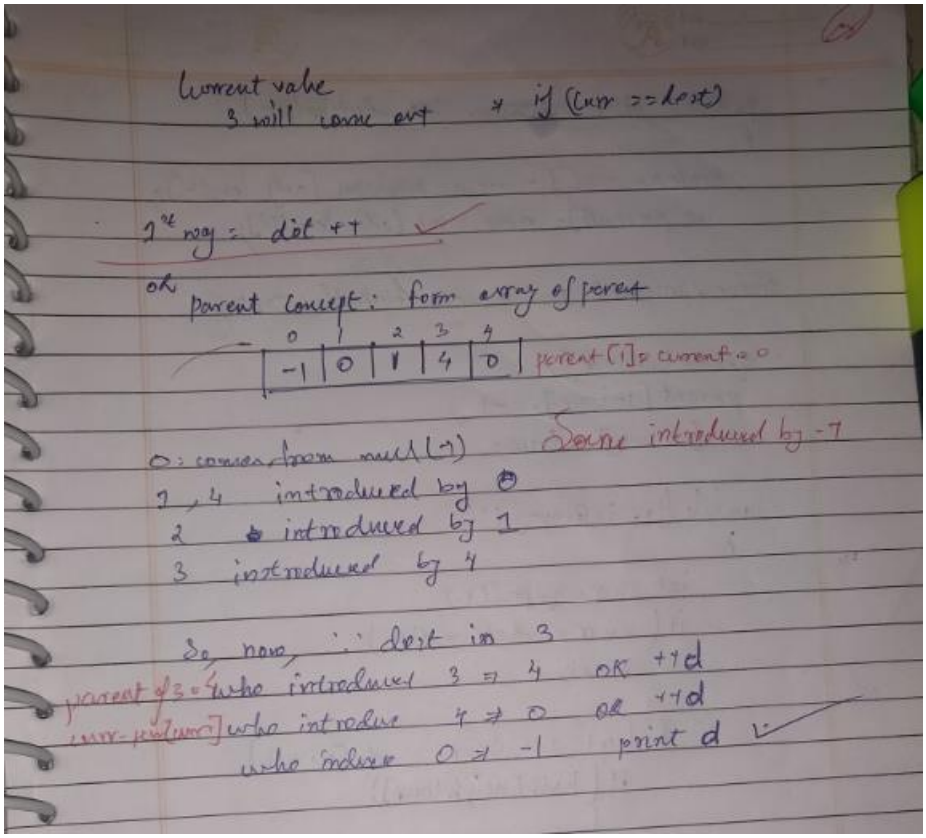
BFS Working explanation and application



Bfs


1
S= 0
2
4
3
Shortest is
0 → 4 → 3
T F F F F
0 1 2 3 4
-1
T
curr
DS used: Queue FIFO
0
pull 0 out & insert connected one
mark 0, 1, 4 visited
1 4
∴ FIFO, 1 will come out first
& neighbour of 1 → 0 but visited
→ 2 not visited, insert 2
& mark visited
4 2
4 will come out
& neighbour of 4 → 0 → visited
→ 3 → not visited & insert 3
& mark visited
2 3
2 will come out
Neighbour of 2 → 1, 3 visited
1
3

Bfs



Output:

```
33  public int bfs( int source , int destination )
34  {
35      boolean vis[]= new boolean[adj.length];
36      int parent[] = new int[adj.length];
37      Queue<Integer> q = new LinkedList<>();
38
39      q.add(source);
40      parent[source]  = -1;
41      vis[source] = true;
42
43      while(!q.isEmpty())
44      {
45          int curr = q.poll();
46          if (curr == destination)
47          break;
48      //else
49          for(int neighbour : adj[curr])  // this will give a LL , of all the possi
50              if(!vis[neighbour])         // if neighbour not visited, mark it visi
51                  {
52                      vis[neighbour] = true;
53                      q.add(neighbour);
54                      parent[neighbour] =  curr;   // and is introduced by curr(the
55                  }
56      }
57      // printing the distance form dest to source.
58      int curr = destination;
59      int distance = 0;
60
61
62      while( parent[curr] != -1 )
63          {
64              System.out.print(curr + "->");
65          curr = parent[curr];
66          distance++;
67          }
68      return distance;
69
```

```
<terminated> Graph [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe
enter the number of vertice and edges
5 6
enter 6 edges
0 1
1 2
2 3
3 4
4 0
4 2
Enter source and destination
0 3
3->4->min distance is 2
```