Breadth First Search

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <queue>
#define LL long long
#define pb push_back

using namespace std;
typedef vector<long long> vi;
typedef vector< vector<long long> > vii;
const LL inf = 1LL << 60;
void makeGraph(vii& G) {
   G[1].pb(2);
   G[1].pb(7);
   G[2].pb(4);
   G[3].pb(2);
   G[3].pb(5);
   G[4].pb(3);
   G[4].pb(6);
   G[5].pb(4);
   G[6].pb(5);
   G[7].pb(8);
}

const int WHITE = 0;
const int GRAY = 1;
const int BLACK = 2;
```

```
pair<vi,vi> bfs(vii& G, LL source)
{
   LL N = G.size();
   vi colour(N,WHITE);
   vi d(N, inf);
   vi p(N, -1);

   colour[source] = GRAY;
   d[source] = 0;
   std::queue<LL> Q;
   Q.push(source);
   while(Q.empty() == false) {
      LL parent = Q.front();
      Q.pop();
      for(LL i=0; i<G[parent].size(); i++) {
         LL child = G[parent][i];
         if (colour[child] == WHITE) {
            colour[child] = GRAY;
            d[child] = d[parent] + 1;
            p[child] = parent;
            Q.push(child);
         }
      }
      colour[parent] = BLACK;
   }
   return make_pair(d,p);
}

int main()
{
   LL n = 9;
   vii G(n); //Graph
   makeGraph(G);
   cout<<G;

   pair<vi,vi> res = bfs(G,1);
   vi d = res.first;
   vi p = res.second;
   LL N = G.size();
   cout<<"Node  d[node]    p[node]\n";
   for(LL i=1; i<N; i++) {
      cout<<"  "<<i<<":\t"<<d[i]<<"        "<<p[i]<<endl;
   }
   return 0;
}
```

Output:

```
D:\Code\ada\bfs\bfs\bin\Debug\bfs.exe
0:

1: 2 7

2: 4

3: 2 5

4: 3 6

5: 4

6: 5

7: 8

8:

Node  d[node]    p[node]
  1:    0         -1
  2:    1          1
  3:    3          4
  4:    2          2
  5:    4          3
  6:    3          4
  7:    1          1
  8:    2          7

Process returned 0 (0x0)   execution time : 0.026 s
Press any key to continue.
```