**Adjacency matrix**

Each cell aij of an adjacency matrix contains **0**, if there is an edge between i-th and j-th vertices, and **1**otherwise. Before discussing the advantages and disadvantages of this kind of representation, let us see an example.

|  |  |
| --- | --- |
| Graph sample | Adjacency matrix for the graph |
| Graph | Adjacency matrix |

|  |  |
| --- | --- |
| Edge (2, 5) | Cells for edge (2, 5) |
| Edge (2, 5) | Cells for the edge (2, 5) |

|  |  |
| --- | --- |
| Graph sample | Adjacency matrix for the graph |
| Edge (1, 3) | Cells for the edge (1, 3) |

**Code snippets**

class Graph {

private:

      bool\*\* adjacencyMatrix;

      int vertexCount;

public:

      Graph(int vertexCount) {

            this->vertexCount = vertexCount;

            adjacencyMatrix = new bool\*[vertexCount];

            for (int i = 0; i < vertexCount; i++) {

                  adjacencyMatrix[i] = new bool[vertexCount];

                  for (int j = 0; j < vertexCount; j++)

                        adjacencyMatrix[i][j] = false;

            }

      }

      void addEdge(int i, int j) {

            if (i >= 0 && i < vertexCount && j > 0 && j < vertexCount) {

                  adjacencyMatrix[i][j] = true;

                  adjacencyMatrix[j][i] = true;

            }

      }

      void removeEdge(int i, int j) {

            if (i >= 0 && i < vertexCount && j > 0 && j < vertexCount) {

                  adjacencyMatrix[i][j] = false;

                  adjacencyMatrix[j][i] = false;

            }

      }

      bool isEdge(int i, int j) {

            if (i >= 0 && i < vertexCount && j > 0 && j < vertexCount)

                  return adjacencyMatrix[i][j];

            else

                  return false;

      }

      ~Graph() {

            for (int i = 0; i < vertexCount; i++)

                  delete[] adjacencyMatrix[i];

            delete[] adjacencyMatrix;

      }

};

Write the code to

a) print all successors of node “3”.

b) print all predecessors of node “3”.

c) Count the number of edges in the graph