IIT Bombay

Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Quiz and Practice Questions on Classes – Part 1

Recap Quiz

Q1. Which of the following are legal structure definitions in C++?

(A) struct T1 {int a; T1 b;};

(B) struct T2 {int a; T2 *b;};

(C) struct T3 {T3 a; T3 *b;};

(D) struct T4 {T4 *a; T4 **b;};

Recap Quiz

Q2. Consider the following code fragment in C++:

struct T {int a; T *next;};

T *x = new T; if (x != NULL) { X1 = 10; X2 = NULL;}

Which of the following choices for X1 and X2 will not give compilation errors?

A. X1: x.a X2: x.next

B. X1: x.a X2: x->next

C. X1: x->a X2: x.next

D. X1: x->a X2: x->next

Recap Quiz

Q3. Consider the following code fragment:

```
struct T1 {char c; int a, b;};
T1 myVar;
```

The number of bytes allocated for myVar on the stack segment is always:

A. At least 9 bytes

B. Exactly 9 bytes

C. Exactly 8 bytes

D. At most 8 bytes

Recap Quiz

Q4. Consider the code fragment

```
struct T1 {int a, b;};
T1 *x = new T1;
```

Assume that "new" successfully allocates an object of type T1. Which of the following accesses member "a" of the dynamically allocated object?

(A) x.a (B) x->a (C) (*x).a (D) (*x)->a

Recap Quiz

IIT Bombay

Q5. Which of the following is/are true of the taxi queue example studied in class:

A. Maximum number of taxis in queue is pre-determined by programmer

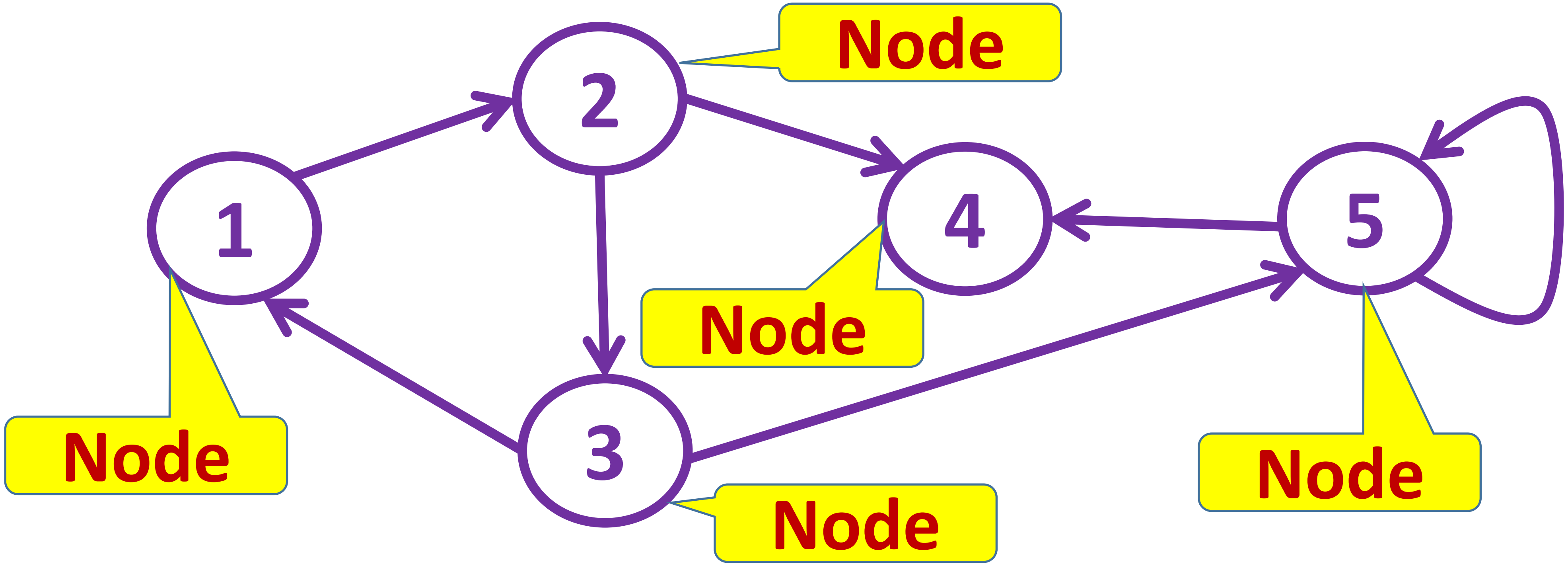
B. Dynamic allocation/de-allocation of structures is used

C. An array of LinkedTaxi objects is used

D. All LinkedTaxi objects are allocated on the heap

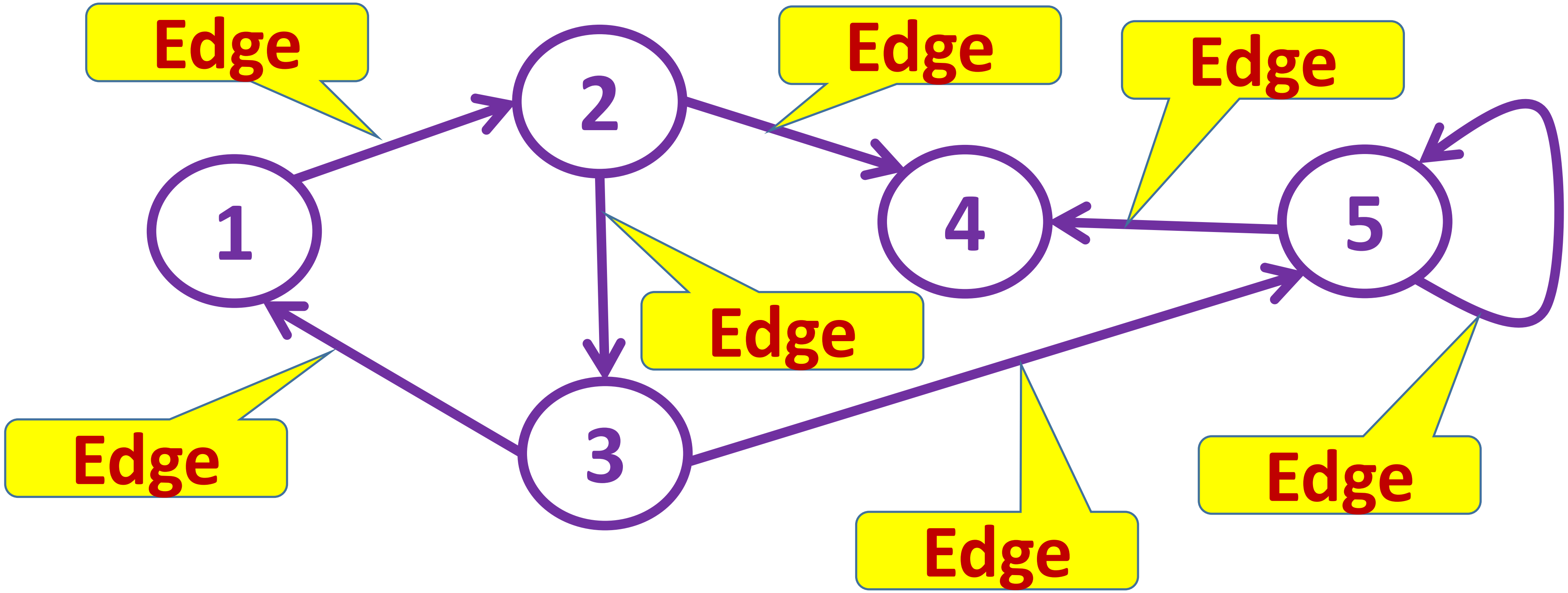
Practice Question 1

A **directed graph** is a finite collection of nodes and directed edges between them



Practice Question 1

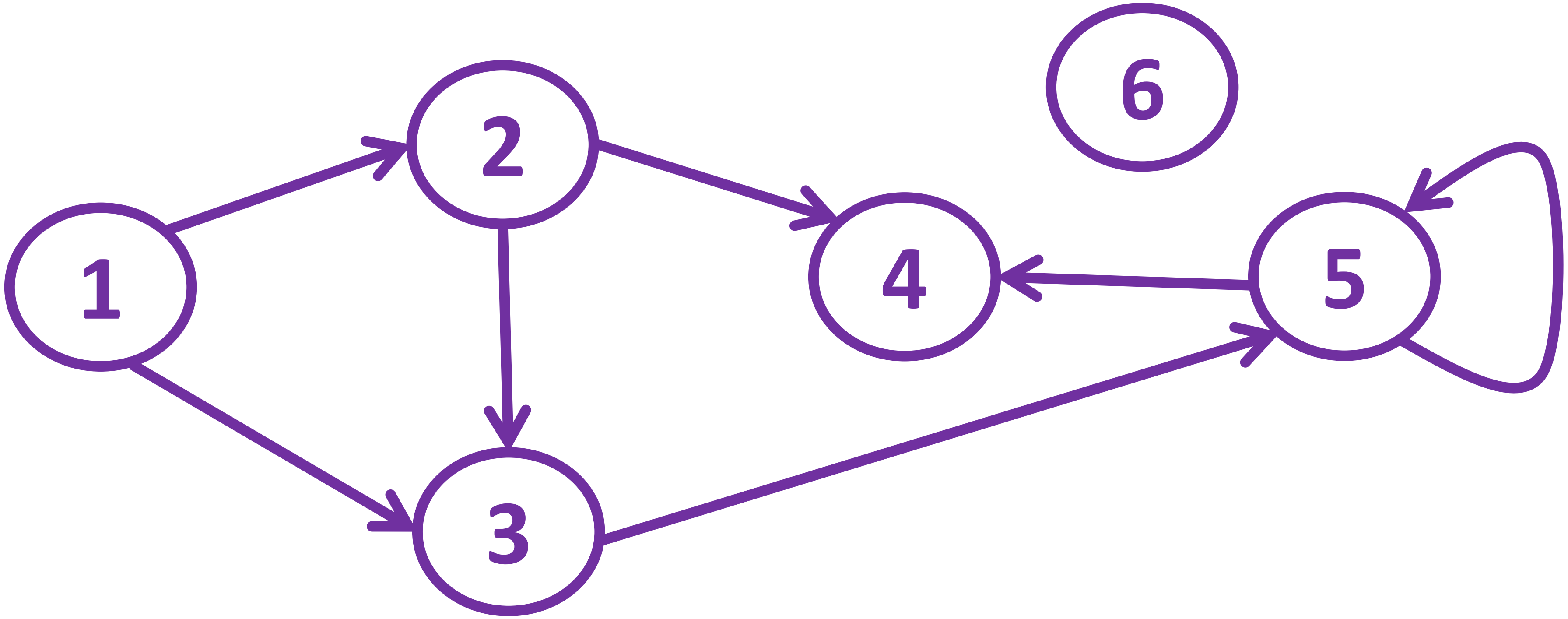
A directed graph is a finite collection of nodes and directed edges between them



Practice Question 1

IIT Bombay

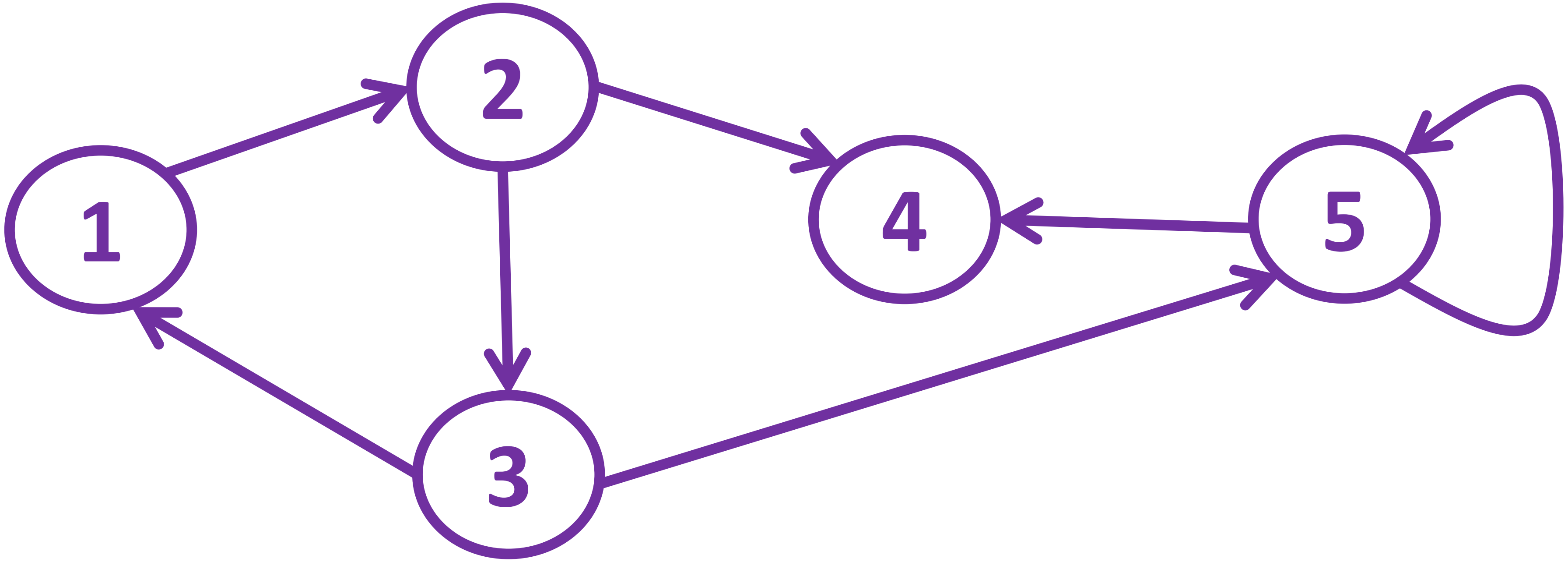
A node can have zero or more incoming/outgoing edges



Practice Question 1

IIT Bombay

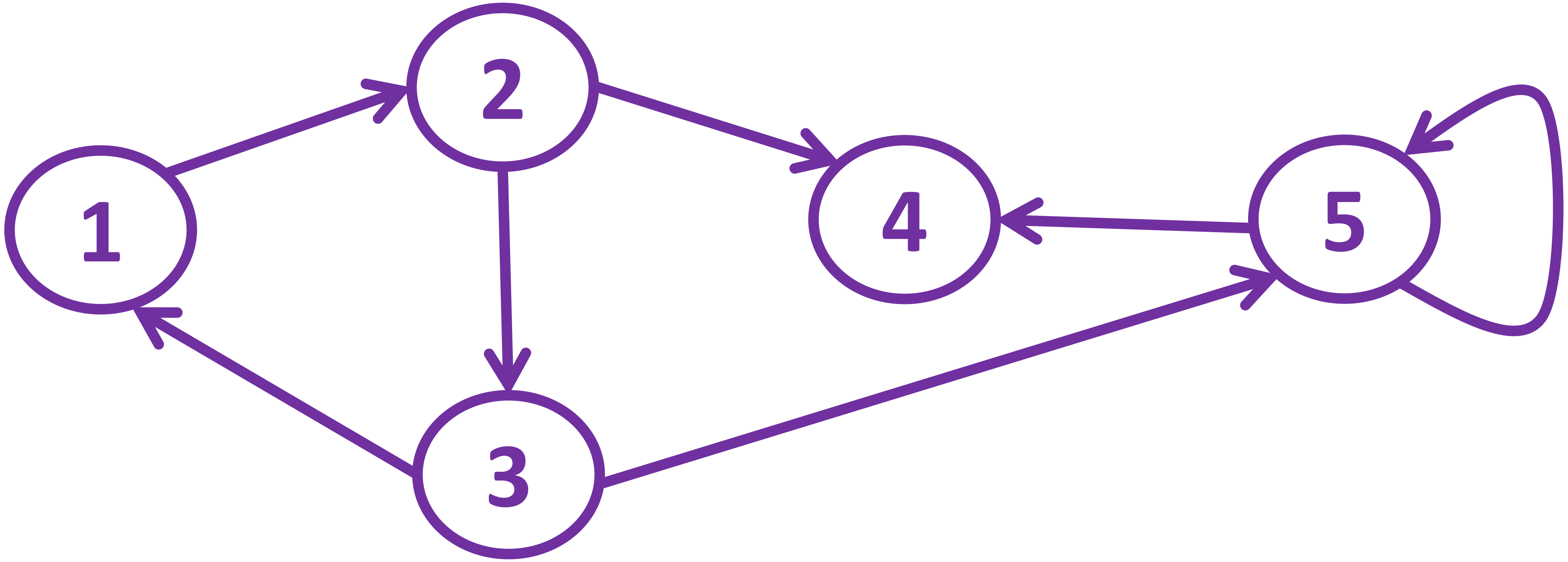
Directed graphs are of central importance in several computational problems



Did you know?

Google treats the web as a directed graph.

“Nodes” are webpages, “edges” are links from one page to another

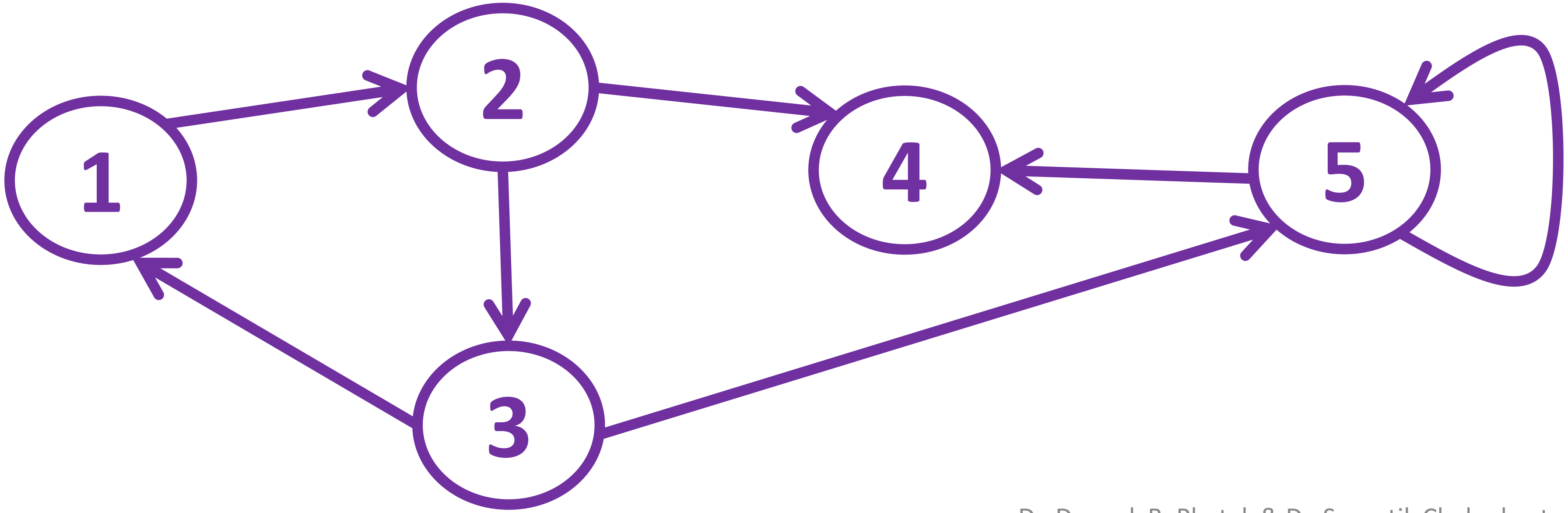


Did you know?

IIT Bombay

Travel portals treat connections between cities as a directed graph.

“Nodes” are cities, “edges” are roads/rail links/air routes



Practice Question 1A

- We want to write a program that reads in information about nodes & edges from the user and constructs a directed graph in memory.
- Nodes must be represented using a structure

```
struct myNode {  …  };
```

- Assume all nodes in the graph are stored in an array named “nodes”. Id of a node is its index in the array.

Practice Question 1A

IIT Bombay

We will use the following structure to represent a node

```
struct myNode {
    int id;
    LinkedNodes *outgoing;
    LinkedNodes *incoming;
}
```

You must decide what this structure should be

Practice Question 1A

```
int main () {
  int numNodes;
  cout << “Give no. of nodes: “;  cin >> numNodes;
  myNode *nodes = new myNode[numNodes];
  if (nodes == NULL) {
    cout << “Memory allocation failure.” << endl;
    return -1;
  }
  else { initNodes(nodes, numNodes); }
```

(continued on next slide …)

Practice Question 1A

```
int  startEdge, endEdge;
while (true) {
    // Reading in edges, one at a time
    cout << “Give start of edge (-1 to quit): “;
    cin >> startEdge;  if (startEdge == -1) break;
    cout << “Give end of edge (-1 to quit): “;
    cin >> endEdge;  if (endEdge == -1) break;
    addEdge(nodes, startEdge, endEdge);
}
```

(continued on next slide …)

Practice Question 1A

```
    // Printing adjacent nodes of every node
    for (int i = 0; i < numNodes; i++) {
       cout << “Nodes with edges from node “ << i << endl;
      printOutNodes(nodes, i);
      cout << “Nodes with edges to node “ << i << endl;
      printInNodes(nodes, i);
   }
 return 0;
}
```

Practice Question 1A

Write the functions

```
void initNodes(myNodes *nodes, int numNodes);
void addEdge (myNodes *nodes, int start, int end);
void printOutNodes(myNodes *nodes, int i);
void printInNodes(myNodes *nodes, int i);
```