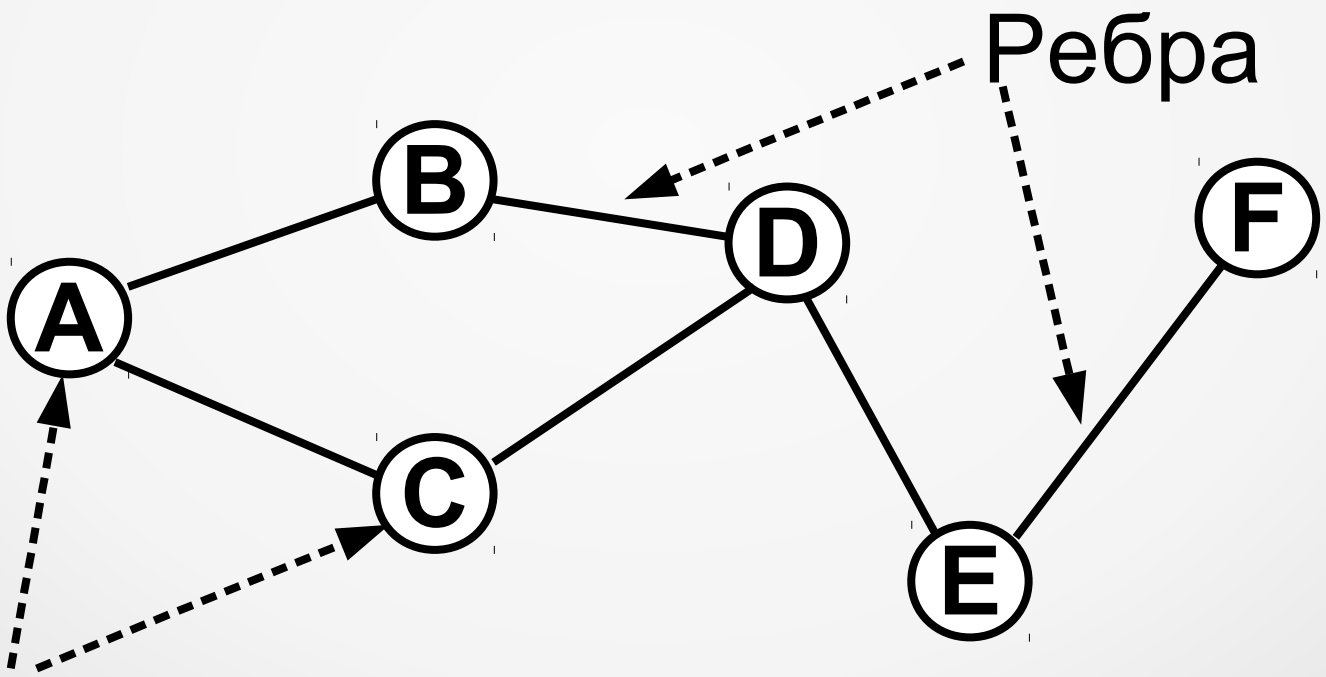
Лекция 6

Алгоритмы на графах

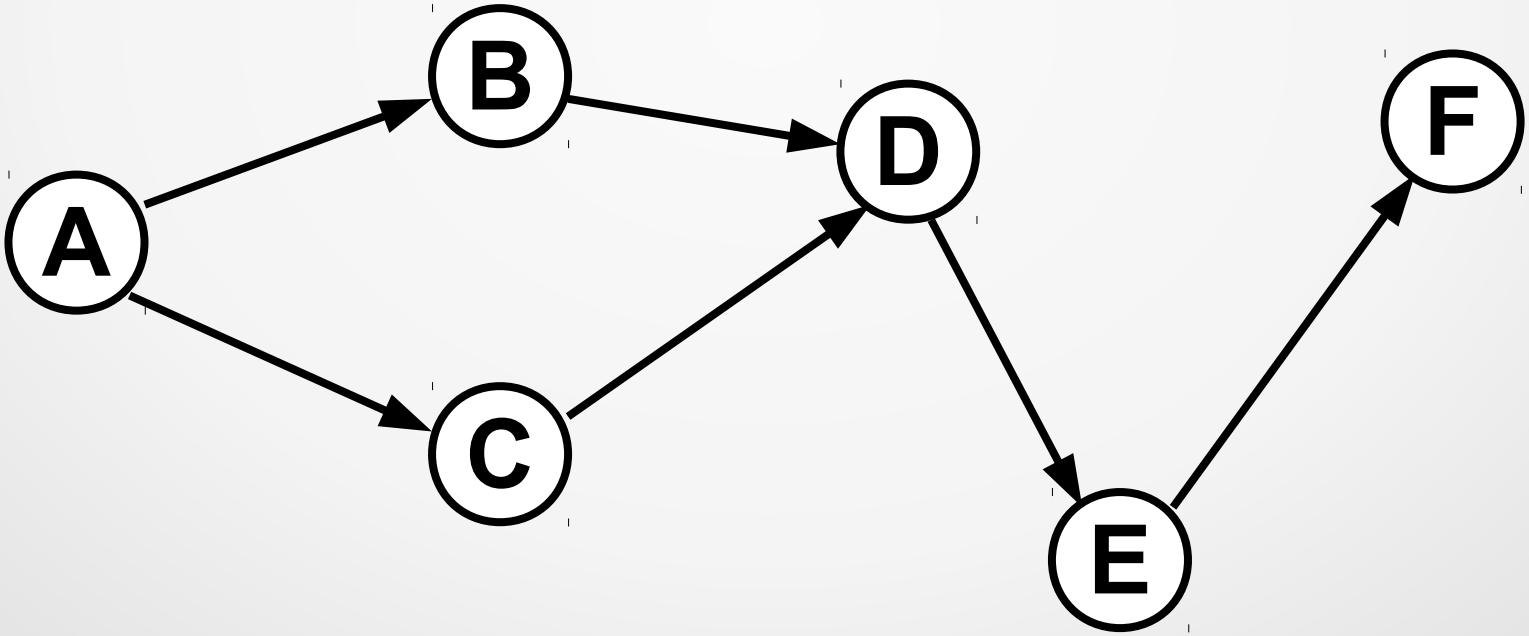
Графы

- Граф — это множество вершин и ребер, связанных между собой



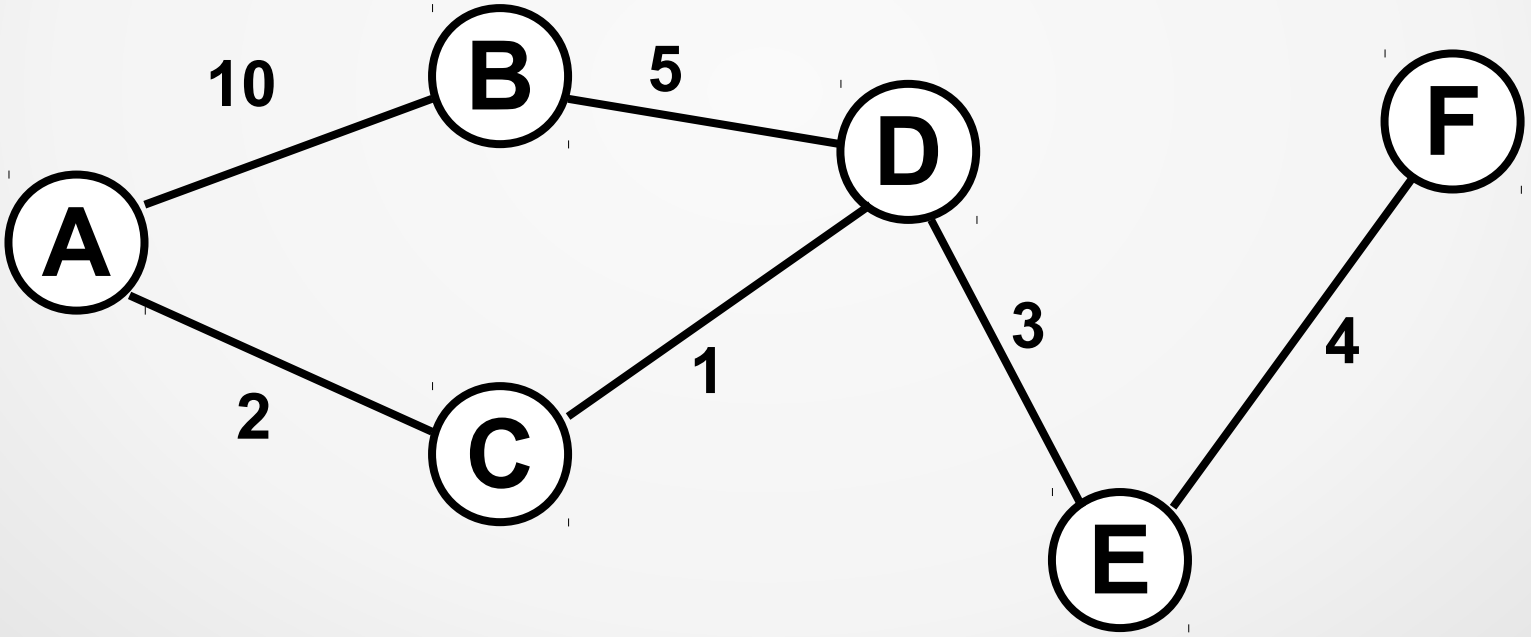
Ориентированный граф

- Ориентированный граф — это граф, в котором для всех ребер определены направления



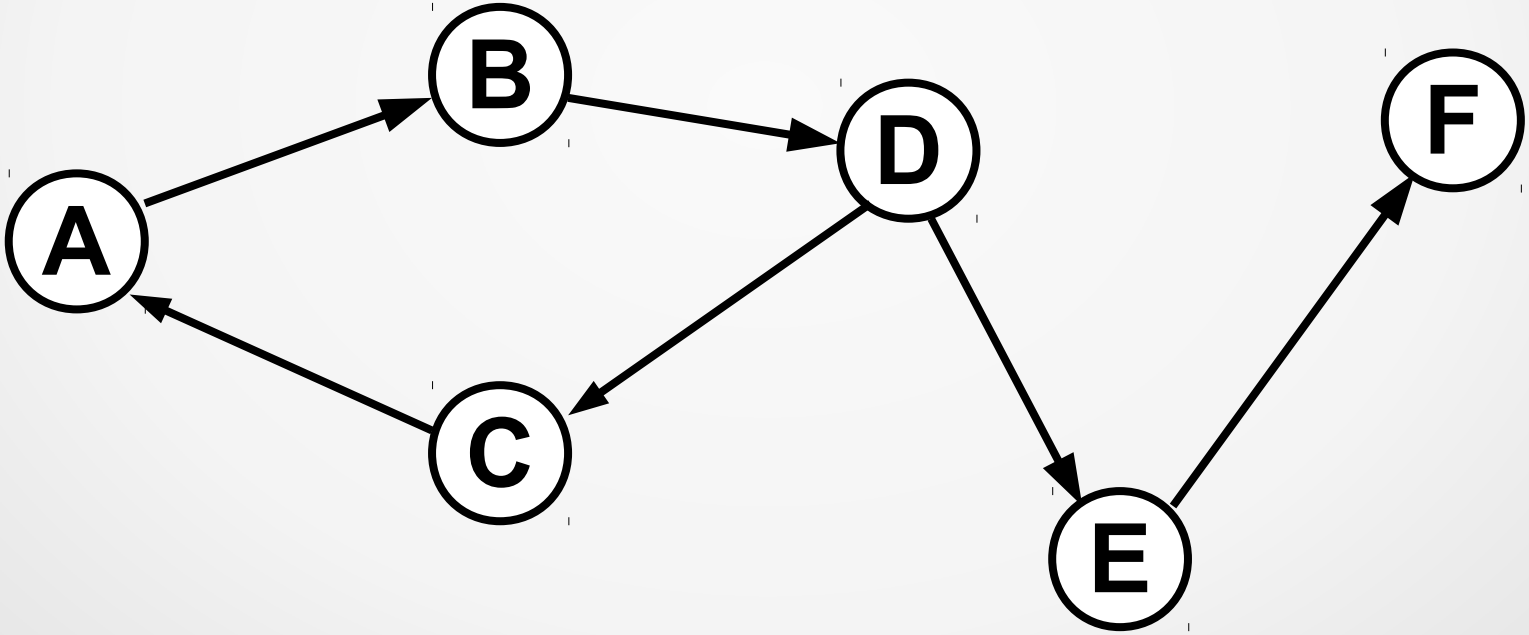
Взвешенный граф

- Взвешенный граф — это граф, в котором у всех ребер есть веса.



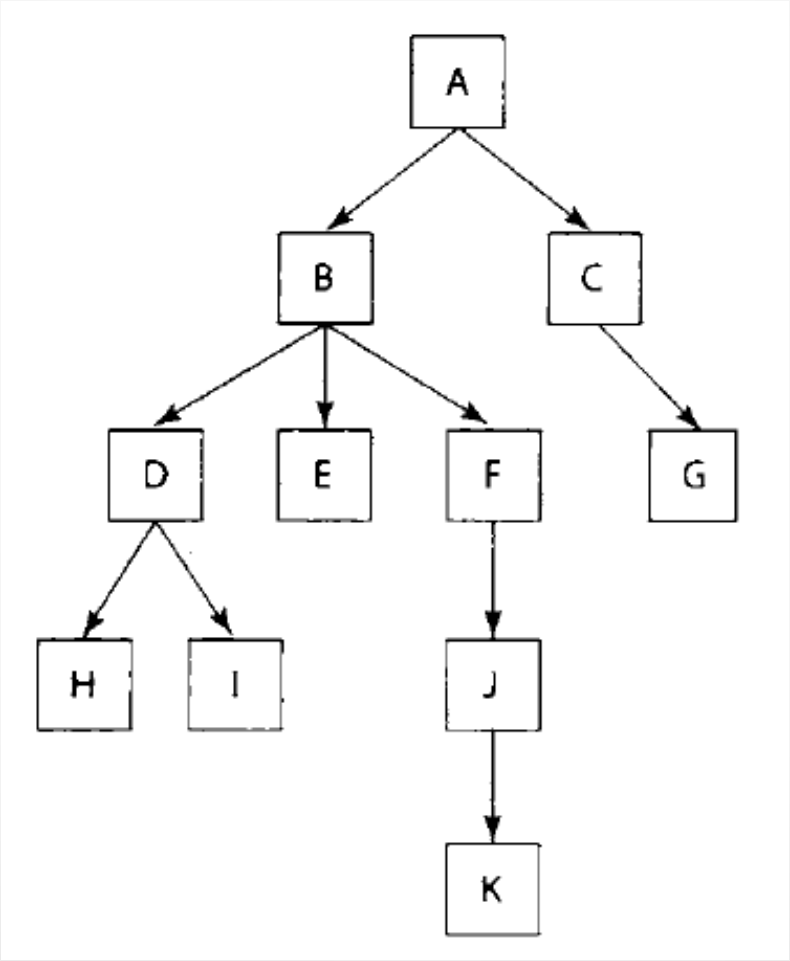
Цикл

- Цикл в графе — это путь ненулевой длины, у которого начальная вершина совпадает с конечной



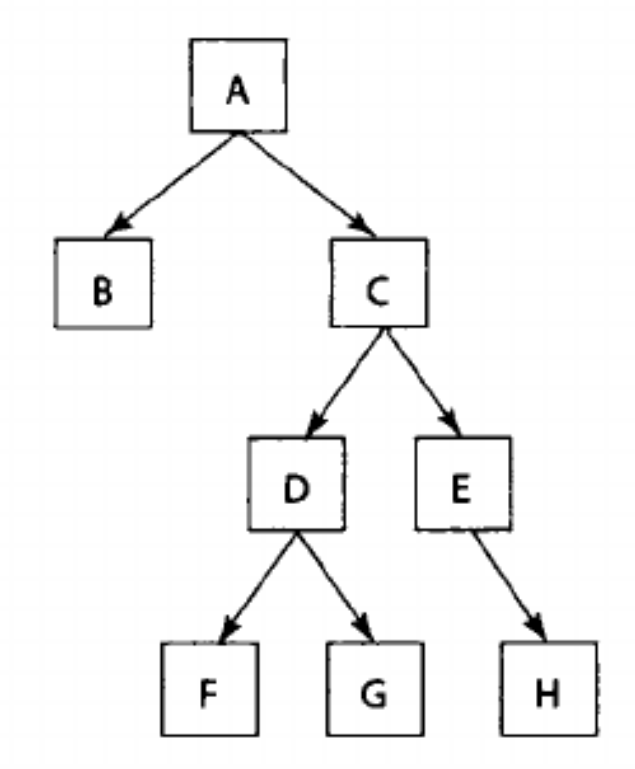
Деревья

- Дерево — **связный** ациклический граф



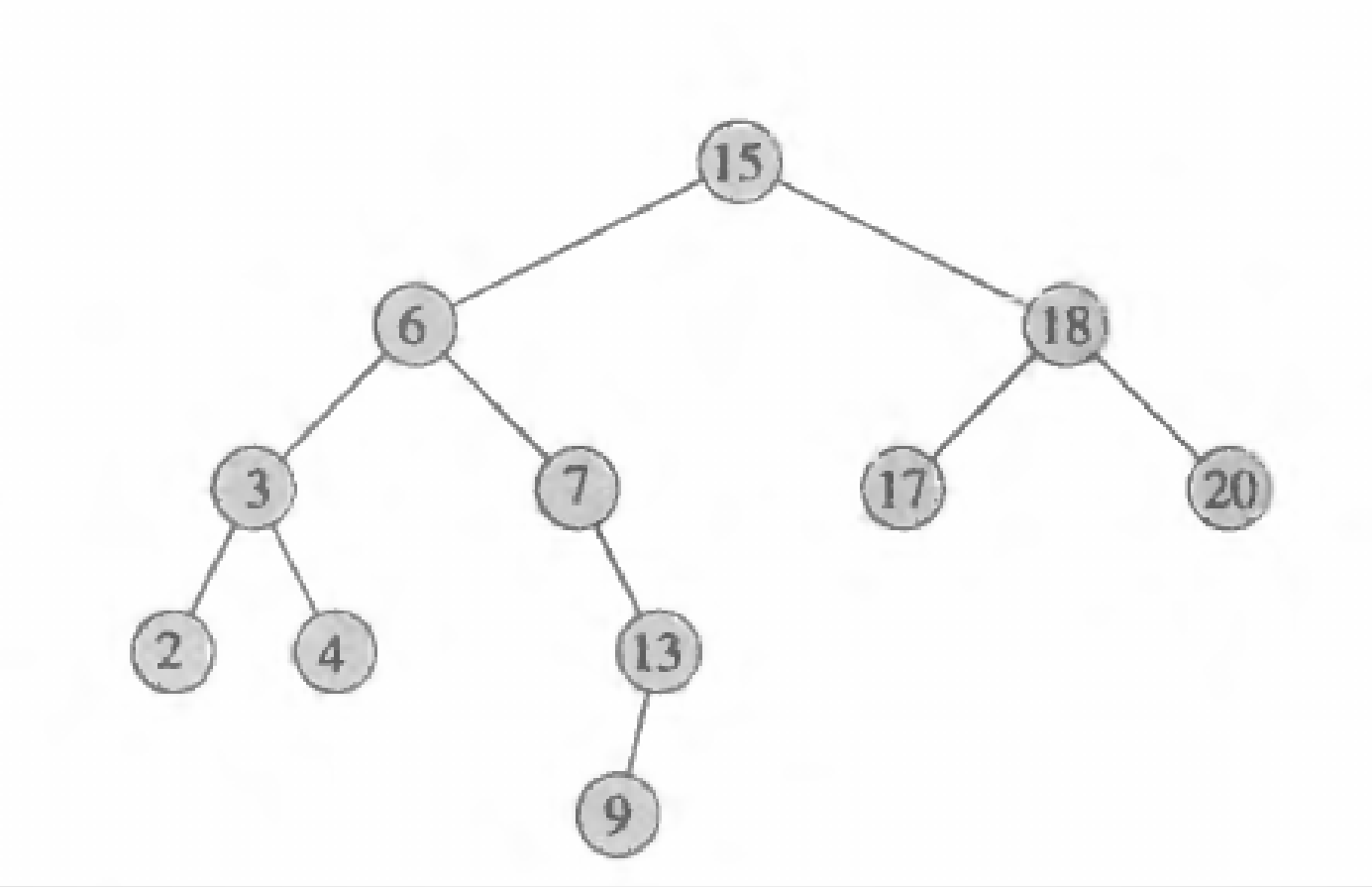
Бинарные деревья

- Каждый элемент бинарного дерева имеет не более двух дочерних элементов. Дочерние элементы называются **левым** и **правым** потомком



Бинарные деревья поиска

- Используются для хранения отсортированных по ключам данных



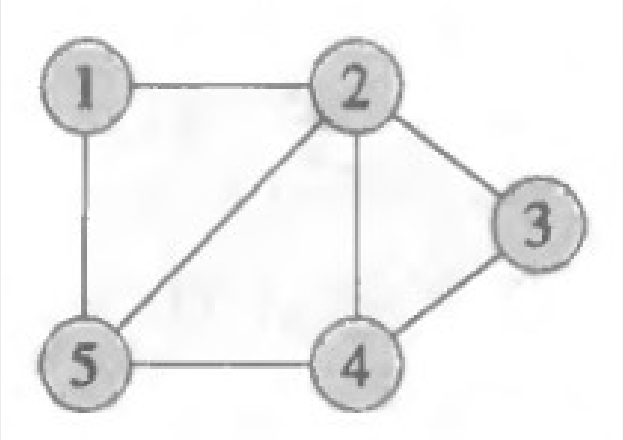
Представление графов

Два стандартных способа:

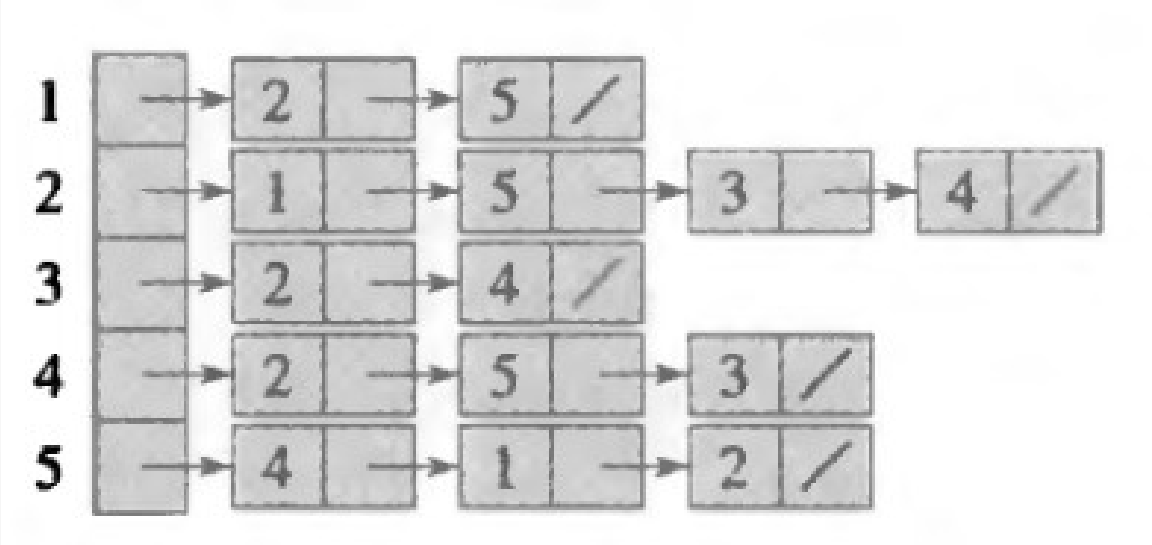
- Набор списков смежных вершин
- Матрица смежности

- Для представления **разреженных** графов обычно применяют списки смежности (первый способ)
- Матрицы смежности предпочтительнее для **плотных** графов, или когда нужно быстро определить, существует ли ребро, соединяющее две заданные вершины (например, в алгоритмах поиска кратчайших путей)

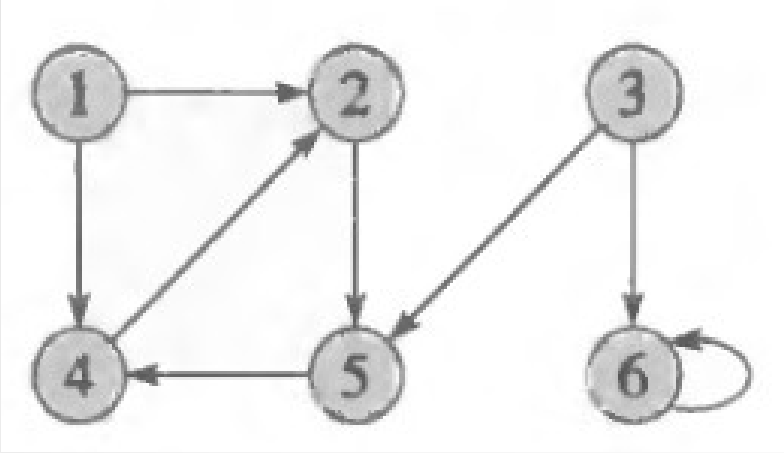
Неориентированный граф

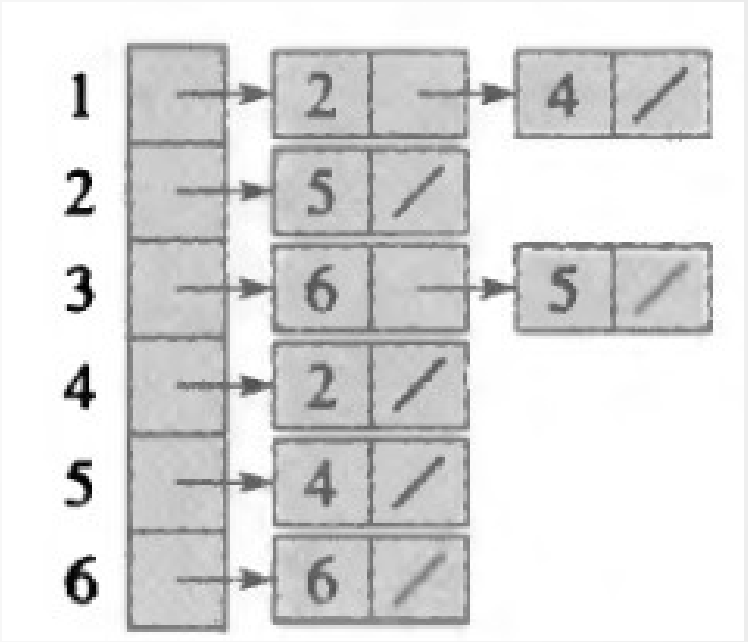


	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0



Ориентированный граф





	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

Поиск в ширину

Исходная вершина — source (s)

- Алгоритм поиска в ширину обходит все рёбра графа G для «открытия» всех вершин, достижимых из s, вычисляя при этом расстояние (минимальное количество рёбер) от s до каждой достижимой из s вершины v.
- Вершины раскрашиваются в белый, серый и чёрный цвета.
- Белые — вершины, в которых мы еще не были
- Серые— просматриваемые вершины
- Черные — просмотренные вершины

Поиск в ширину

```
BFS(G, s)
 1  for Каждой вершины u ∈ G.V − {s}
 2       u.color = WHITE
 3       u.d = ∞
 4       u.π = NIL
 5  s.color = GRAY
 6  s.d = 0
 7  s.π = NIL
 8  Q = ∅
 9  ENQUEUE(Q, s)
10  while Q ≠ ∅
11       u = DEQUEUE(Q)
12       for Каждой вершины v ∈ G.Adj[u]
13            if v.color == WHITE
14                 v.color = GRAY
15                 v.d = u.d + 1
16                 v.π = u
17                 ENQUEUE(Q, v)
18       u.color = BLACK
```

- Переменная s — исходная вершина
- Атрибут d — расстояние от s
- Атрибут π - предшественник

Поиск в глубину

Исходная вершина — source (s)

- Атрибуты d и f — время открытия и закрытия вершины
- Белые — вершины, в которых мы еще не были
- Серые— просматриваемые вершины
- Черные — просмотренные вершины

```
DFS(G)
1  for каждой вершины u ∈ G.V
2      u.color = WHITE
3      u.π = NIL
4  time = 0
5  for каждой вершины u ∈ G.V
6      if u.color == WHITE
7          DFS-VISIT(G, u)

DFS-VISIT(G, u)
 1  time = time + 1                  // Открыта белая вершина u
 2  u.d = time
 3  u.color = GRAY
 4  for каждой v ∈ G.Adj[u]          // Исследование ребра (u, v)
 5      if v.color == WHITE
 6          v.π = u
 7          DFS-VISIT(G, v)
 8  u.color = BLACK                  // Завершение работы с u
 9  time = time + 1
10  u.f = time
```