

---

АДРИАНОВ Н.М.  
ИВАНОВ А.Б.

# АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ

ГРАФЫ. DFS.  
ТОПОЛОГИЧЕСКАЯ СОРТИРОВКА.  
СИЛЬНОСВЯЗАННЫЕ КОМПОНЕНТЫ.

---

# ПРЕДСТАВЛЕНИЕ ГРАФА В ПРОГРАММЕ

**Матрица смежности**

**Списки смежности**

# ПОИСК В ГЛУБИНУ (DFS = DEPTH FIRST SEARCH)

---

```
procedure DFS( G, s, t ):
  if s = t:
    return true
  visited = array of size G.V
  return DFS( G, s, t, visited )
end

procedure DFS( G, s, t, visited ):
  visited[s] = true
  for x in G.neighbours(s):
    if x = t:
      return true
    if not visited[x]:
      if DFS(G, x, t, visited):
        return true
  end
  return false
end
```

---

## DFS БЕЗ РЕКУРСИИ

Заменить в алгоритме BFS очередь на стек!

## ПОЛНЫЙ ОБХОД ВСЕГО ГРАФА (DFS)

УТВ. После вызова `DFS(G, x)`  
для всех вершин `y`, достижимых  
из `x`, `visited[y] = true`

---

```
procedure DFS( G ):
  visited = array bool[G.V]
  for x in G.vertices:
    if not visited[x]:
      DFS(G, x)
  end
end
```

```
procedure DFS( G, x ):
  visited[x] = true
  Previsit(x)
  for y in G.neighbours(x):
    if not visited[y]:
      DFS(G, y)
  end
  Postvisit(x)
end
```

---

# КОМПОНЕНТЫ СВЯЗНОСТИ НЕОРИЕНТ.ГРАФА

---

```
procedure Previsit( x ):
    ccnum[x] = cc
end
```

---

---

```
procedure DFS( G ):
    cc = 0
    ccnum = array int[G.V]
    visited = array bool[G.V]
    for x in G.vertices:
        if not visited[x]:
            DFS(G, x)
            cc = cc + 1
        end
    end
end
```

```
procedure DFS( G, x ):
    visited[x] = true
    Previsit(x)
    for y in G.neighbours(x):
        if not visited[y]:
            DFS(G, y)
        end
    end
    Postvisit(x)
end
```

---

# DFS В ОРИЕНТИРОВАННЫХ ГРАФАХ

---

```
procedure Previsit( x ):
  pre[x] = clock
  clock = clock + 1
end
```

```
procedure Postvisit( x ):
  post[x] = clock
  clock = clock + 1
end
```

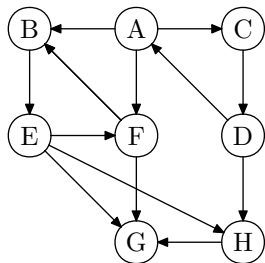
---

---

```
procedure DFS( G ):
  pre = array int[G.V]
  post = array int[G.V]
  clock = 1
  ...
end
```

---

## DFS В ОРИЕНТИРОВАННЫХ ГРАФАХ



A: B,C,F

E: F,G,H

B: E

F: B,G

C: D

G: -

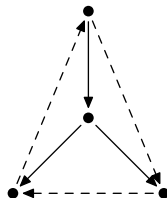
D: A,H

H: G



## ТИПЫ РЕБЕР

древесное/прямое	[	[	]	]
	u	v	v	u
обратное	[	[	]	]
	v	u	u	v
перекрестное	[	]	[	]
	v	v	u	u



УТВ 1. Для любых вершин  $u$  и  $v$  отрезки  $[\text{pre}(u), \text{post}(u)]$  и  $[\text{pre}(v), \text{post}(v)]$  либо вложены, либо не пересекаются.

УТВ 2. Ориентированный граф содержит цикл  $\Leftrightarrow$  существует обратное ребро.

УТВ 3. Для любого ребра  $(u, v)$  в ориентированном ациклическом графе (DAG)  $\text{post}(u) > \text{post}(v)$ .

# ТОПОЛОГИЧЕСКАЯ СОРТИРОВКА

**Дано:** Ориентированный ациклический граф  $G = (V, E)$ .

**Надо:** Пронумеровать вершины  $V$  графа так, чтобы для любого ребра  $(u, v) \in E$  номер вершины  $v$  был больше номера вершины  $u$ .

*(Другими словами: расположить вершины на временной оси так, чтобы все ребра были направлены “в будущее”).*

**Решение:** Выводим вершины в обратном post-порядке.

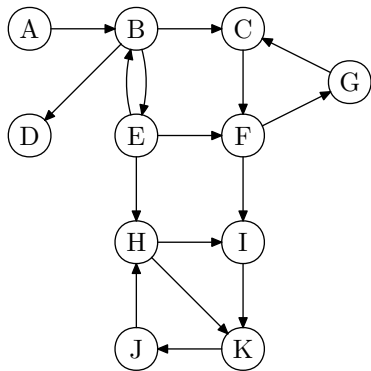
# ИСТОКИ И СТОКИ В DAG

Вершина называется *исток*ом, если не существует входящих в нее ребер.

Вершина называется *сток*ом, если не существует исходящих из нее ребер.

Утв. В ориентированном ациклическом графе (DAG) существует хотя бы один исток и хотя бы один сток.

# КОМПОНЕНТЫ СИЛЬНОЙ СВЯЗНОСТИ



# КОМПОНЕНТЫ СИЛЬНОЙ СВЯЗНОСТИ

УТВ 1. Пусть  $v$  лежит в компоненте-стоке. Тогда  $\text{DFS}(G, v)$  обойдет все вершины в компоненте и только их.

## КОМПОНЕНТЫ СИЛЬНОЙ СВЯЗНОСТИ

УТВ 2. Пусть  $C$  и  $C'$  – компоненты сильной связности, и существует ребро из  $C$  в  $C'$ . Тогда

$$\max_{c \in C} \text{post}(c) > \max_{c' \in C'} \text{post}(c')$$

СЛЕДСТВИЕ. Вершина с максимальным post-значением лежит в компоненте-истоке.

# АЛГОРИТМ КОСАРАЙЮ

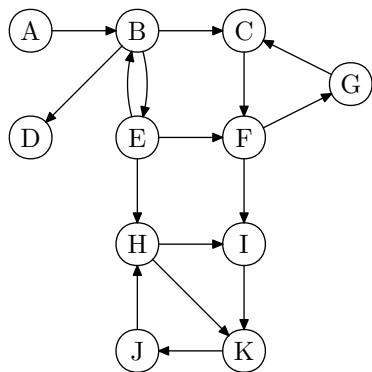
(S. Rao Kosaraju, Micha Sharir, 1979)

$G^R$  – обращенный граф

1. Запустить DFS для  $G^R$ .
2. Запустить алгоритм нахождения компонент  $G$  (как в неориентированном случае) в обратном post-порядке.

# АЛГОРИТМ ТАРЬЯНА

Robert Tarjan – алгоритм с однократным DFS, 1972



index, lowlink



## 2-ВЫПОЛНИМОСТЬ (2-SAT)

**Дано:** булева формула в КНФ такая, что в каждом терме только 2 литерала:

$$F(x_1, x_2, \dots, x_n) = \bigwedge_{\alpha} (z_{\alpha} \vee w_{\alpha}), \quad \text{где } z_{\alpha}, w_{\alpha} = x_i \text{ или } \bar{x}_i$$

**Надо:** Найти такой набор значений переменных, чтобы формула приняла значение “истина”.