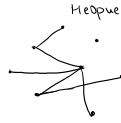
Графы

$G=(V,E)$ — вершины, рёбра

Неориент. граф

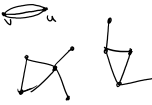


Ор. граф

Неор. графы

Простой граф: без петель и кратных рёбер

Компонента связности: v связ с u — $\exists$ путь $v \to u$



Хранение графов

① Матрица смежности

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

Зам. Граф неор. $\Rightarrow$ M симметр.

$Mem = O(V^2)$

проверка на смежность v и u: $O(1)$

перебрать соседей v: $O(n)$

② Списки смежности $edges[v] = [\dots]$

1: [2]
2: [1, 4, 3]
3: [2, 4]
4: [2, 3]

неор. граф: $\deg v$ – степень

ор. граф: $\deg_{in} v$ – вход. степень, $\deg_{out} v$ – исх. степень

$e=(v,u)$ неор.: $edges[v].append(u)$, $edges[u].append(v)$

ор.: $edges[v].append(u)$

$Mem = O(E)$

проверка на смежность v и u: $\min(\underbrace{len(edges[v])}_{\deg v}, len(edges[u])) \le$

$\le O(V)$

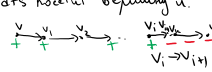
перебрать соседей v: $O(\deg v)$

Поиск в глубину (depth-first search) DFS

```
used=[False]*V
def dfs(v):
    used[v]=True
    for u in edges[v]:
        if not used[u]:
            dfs(u)
```

Утв. Запустим dfs в вершине v. Существует путь $v \to u$ по непосещ. вершинам. Тогда dfs посетит вершину u.

Д-во



■

Задача Граф неор., не обяз. связный. Нужно посчитать #комп. св-ти.

```
def dfs(v):
    used[v]=True
    s=1
    for u in edges[v]:
        if not used[u]:
            s+=dfs(u)
    return s
def main():
    cnt=0
    for v in range(V):
        if not used[v]:
            print(dfs(v))
            cnt+=1
    print(cnt)
```

$T(V,E) = O(V+E)$ – со списками смежности

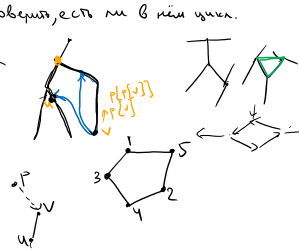
$M(V,E) = O(V)$

А теперь раскрасим компоненты.

$T(V,E) = O(V^2)$ – с матрицей смежности

Задача Дан неориент. граф. Проверить, есть ли в нём цикл.

```
def dfs(v,p=-1):
    used[v]=True
    for u in edges[v]:
        if not used[u]:
            dfs(u,v)
        elif u!=p:
            print("cycle!")
            exit(0)
```



Задача Есть ли цикл в орграфе? (digraph)

```
used=[0]*V; s=[]
def dfs(v):
    used[v]=1; s.append(v)
    for u in edges[v]:
        if used[u]==0:
            dfs(u)
        elif used[u]==1:
            print("Cycle!")
            while s[-1]!=u:
                print(s[-1])
                s.pop()
            print(u)
    used[v]=2; s.pop()
```

0 цвет – белые
1 цвет – серые
2 цвет – чёрные

(ответ переворачут!)