A2

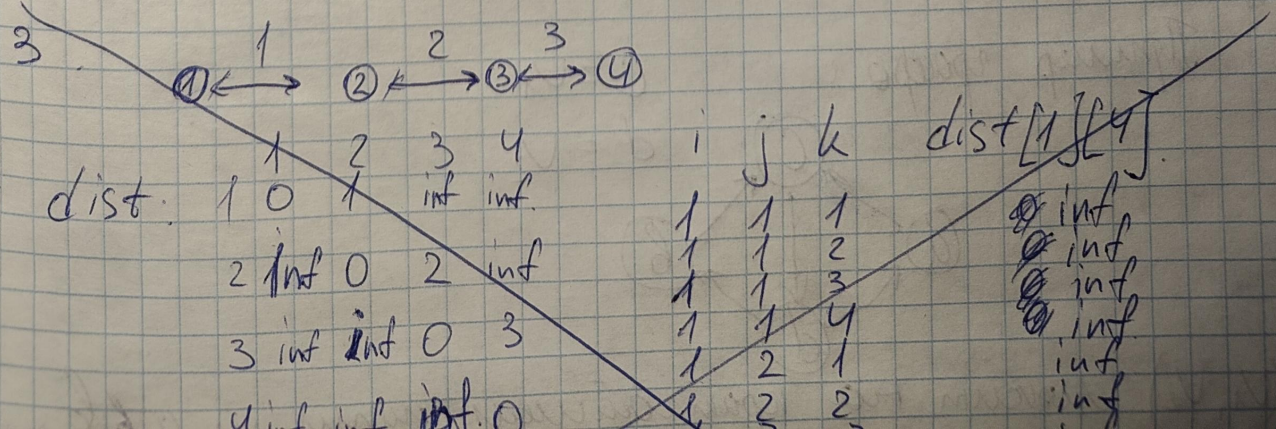
1. Алгоритм изменится лишь в том, что все операции сложения поменяются на умножение:

```
if (dist[u] * weight(u,v) < dist[v])
    dist[v] = dist[u] * weight(u,v)
```

Алгоритм будет работать корректно, если длины ребер заданы вещественными числами $\in [1; +\infty)$. Если будут отрицательные числа, алгоритм будет работать непредсказуемо. Если будут веса в интервале $[0, 1)$, при прохождении через такие ребра путь будет сокращаться.

2. Код приведен в приложенном файле.

Невозможно будет восстановить граф, если матрица задана некорректно или если в матрице присутствуют ~~длины~~ ~~дуги~~ ~~бесконечности~~ длины со значением $-\infty$.



3. ① $\xrightarrow{1}$ ③ $\xrightarrow{2}$ ④ $\xrightarrow{3}$ ②

dist	1	2	3	4
1	0	inf	1	inf
2	inf	0	inf	inf
3	inf	inf	0	2
4	inf	3	inf	0

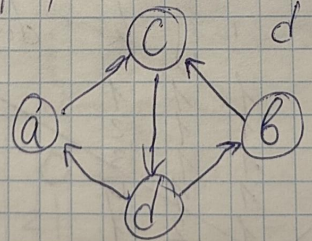
По графу видим, что кратчайший путь от 1 до 2 равен 6. Изначально этот путь равен inf. Улучшение может произойти, только если $i=1$, $j=2$:

i	j	k	dist[1][2]
1	1	1	inf
1	1	2	inf
1	1	3	inf
1	1	4	inf
1	2	1	inf
1	2	2	inf
1	2	3	inf
1	2	4	inf
1	3	1	inf

Так как больше в алгоритме не будет попыток улучшить путь от 1 к 2, он останется равным inf

$\Rightarrow$ Алгоритм работает неправильно.

4. Пример графа:



$C = V_1$

$d = V_2$

(V_1, V_2) лежит на кратчайшем пути из a в в. (единственном)

(V_1, V_2) — и — из в в a.

Про такие графы можно сказать:

1) Есть цикл, содержащий v_1, v_2, b, т.к. есть путь из b в v_1 и есть путь из v_1 через v_2 в b.

2) Аналогично есть цикл, содержащий v_1, v_2, a.

Ограничения на ~~все~~ алгоритмы могут возникнуть, если в ~~одном цикле в графе~~ графе хотя бы один цикл отрицательной длины. В таком случае будет нельзя применить алгоритм Дейкстры.

А3.