Содержание

1 Условие задания

Робот блуждает лабиринте. У робота есть исходная точка, в которой он начинает путешествие, конечная точка и начальный заряд батареи (количество ходов). Лабиринт описывается ориентированным графом и может изменяться после каждого хода робота.

На вход подаются описание лабиринта, описание робота и сценарий изменения лабиринта. Требуется проверить, может ли робот достичь конечной точки.

1.1 Общие требования

Решение задачи — это исходный код, содержащий средства автоматической сборки исполняемого файла `labyrinth`.

Исполняемый файл `labyrinth` должен завершаться на корректных данных, возвращая значение "0", если робот может достичь конечной точки, и "-1", если не может. Для простоты "защита от дурака" (обработка некорректных данных) не включена в обязательные требования.

Данные берутся из трёх файлов. При запуске `labyrinth` без параметров данные должны считываться из файлов

- `labyrinth.txt` — описание лабиринта,
- `robot.txt` — описание исходной и конечной точек робота и его начального заряда,
- `changes.txt` — описание изменений, происходящих в лабиринте

Файл `labyrinth` должен иметь возможность запуска со следующими опциями:

- `-l` *`имя_файла`* — описание лабиринта считывается из указанного файла, а не файла по умолчанию;
- `-r` *`имя_файла`* — описание робота считывается из указанного файла, а не файла по умолчанию;
- `-c` *`имя_файла`* — описание изменений лабиринта считывается из указанного файла, а не файла по умолчанию.

1.2 Описание лабиринта

Лабиринт — это ориентированный граф следующего вида:

- вершины — это целые числа от 0 до $N-1$, $N \geq 1$;
- каждая вершина v помечена числом $\tau(v)$ от 0 до $VT-1$, $VT \geq 1$;
- каждая дуга e помечена числом $\rho(e)$ от 0 до $ET-1$, $ET \geq 1$.

M — это количество дуг графа, $M \geq 1$.
Формат описания лабиринта:

- Первая строка — 4 числа, разделённых пробелами: N, M, VT, ET.
- Вторая строка — числа от 0 до $VT-1$, разделённые пробелами, — метки вершин графа в порядке нумерации.
- После первых двух строк следует ещё ровно M строк, в каждой из них записаны 3 числа, разделённых пробелами:

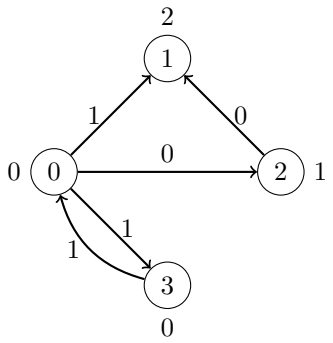
– вершина, из которой исходит дуга,

– метка дуги,

– вершина, в которую ведёт дуга.

Пример:

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



(*иллюстрация графа приведена для наглядности, на вход подаётся только куча чисел, записанная слева*)

1.3 Описание робота

Описание робота — это одна строка, содержащая 3 числа, разделённые пробелами:

- исходная точка (вершина лабиринта, в которой робот начинает путешествие),
- конечная точка (вершина, достижимость которой требуется проверить),
- заряд батареи (целое неотрицательное число S — количество шагов, производимых роботом)

1.4 Описание изменений

Допустимы следующие команды изменений лабиринта:

- `nop`
  - лабиринт не изменяется;
- `edges off E`
  - `E` — метка дуг (*число заданного диапазона, как и далее в аналогичных случаях*);
  - дуги с меткой `E` блокируются;
- `edges on E`
  - `E` — метка дуг;
  - дуги с меткой `E` разблокируются, если были заблокированы;
- `nodes off V`
  - `V` — метка вершин;
  - вершины с меткой `V` блокируются;
- `nodes on V`
  - `V` — метка вершин;
  - вершины с меткой `V` разблокируются;
- `reverse all`
  - все дуги лабиринта разворачиваются (начало дуги становится концом, а конец — началом);
- `reverse E`
  - `E` — метка дуг;
  - дуги с меткой `E` разворачиваются.

Файл описания изменений содержит произвольное число строк, по одной команде в каждой строке. Если строк с изменениями лабиринта меньше, чем S, то в недостающих строках подразумевается команда `nop`.

1.5 Как ходит робот

Начальное положение робота — это вершина лабиринта, равная начальной точке робота.

Ходы робота и изменения лабиринта соотносятся следующим образом:

- Применяется первое изменение лабиринта.
- Робот переходит произвольно по одной дуге.
- Применяется второе изменение лабиринта.
- Робот переходит произвольно по одной дуге.
- ...

Робот совершает столько переходов, каков заряд батареи S.

Робот **не может** переходить по заблокированным дугам и в заблокированные вершины.

Если робот попал в вершину, не являющуюся конечной точкой, и эта вершина заблокировалась, то он **взрывается** (не достигает конечной точки). Если робот попал в конечную точку, то это безусловно **успешное** достижение точки, даже если он немедленно после этого взрывается при блокировании вершины.

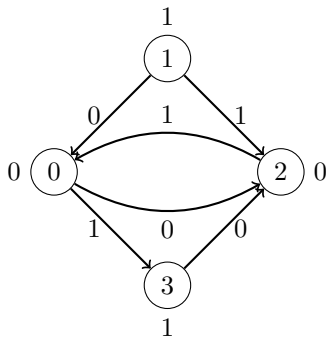
На каждом шаге робот **обязан** сделать переход по дуге (не умеет тормозить). Если исходящих дуг нет, то робот врезается в стенку и **взрывается**.

1.6 Примеры входных данных

1.6.1 Успешное путешествие

Описание лабиринта:

```
4 6 2 2
0 1 0 1
0 0 2
0 1 3
1 0 0
1 1 2
2 1 0
3 0 2
```



Описание робота:

```
0 1 8
```

Описание изменений:

```
edges off 0
edges on 0
reverse 0
reverse all
nodes off 1
nodes on 1
```

Возможное успешное путешествие робота: $0 \to 3 \to 2 \to 3 \to 0 \to 2 \to 1$

1.6.2 Несуществование успешного путешествия

Примеры изменения, которое достаточно внести в данные предыдущего примера, чтобы успешное путешествие перестало существовать:

1. Заряд — 5.
2. Первая команда — `nodes off 0`.
3. Четвёртая команда — `nodes off 0`.
4. Шестая команда — `nop`.

1.7 Оценка задания

Возможны три варианта реализации решения:

1. Реализация всех частей системы как булевых функций, и решения задачи как преобразования булевых функций (**оценивается высоко**).
2. Не задействующее аппарат булевых функций, но основанное на `stl/boost`-реализации **всех** частей задачи, подходящих для реализации этих частей `stl/boost`-контейнерами, и *грамотном* оперировании этими контейнерами (**оценивается средне**).
3. Все остальные варианты реализации (**по умолчанию оцениваются низко**).

Максимальная оценка за задание: **120 баллов**.

1.7.1 Реализация, основанная на булевых функциях

- **40 баллов:** решение с поддержкой команды `nop`.
- **10 баллов:** поддержка команды `edges off E`.
- **10 баллов:** поддержка команды `edges on E` при наличии команды `edges off E`.
- **10 баллов:** поддержка команды `nodes off V`.
- **10 баллов:** поддержка команды `nodes on V` при наличии команды `nodes off E`.
- **20 баллов:** поддержка команды `reverse all`.
- **20 баллов:** поддержка команды `reverse E`.

1.7.2 Реализация, основанная на `stl/boost`

- **30 баллов:** решение с поддержкой команды `nop`.
- **7.5 баллов:** поддержка команды `edges off E`.
- **7.5 баллов:** поддержка команды `edges on E` при наличии команды `edges off E`.
- **7.5 баллов:** поддержка команды `nodes off V`.
- **7.5 баллов:** поддержка команды `nodes on V` при наличии команды `nodes off E`.
- **15 баллов:** поддержка команды `reverse all`.
- **15 баллов:** поддержка команды `reverse E`.

1.7.3 Другие реализации

- **20 баллов:** решение с поддержкой команды `nop`.
- **5 баллов:** поддержка команды `edges off E`.
- **5 баллов:** поддержка команды `edges on E` при наличии команды `edges off E`.
- **5 баллов:** поддержка команды `nodes off V`.
- **5 баллов:** поддержка команды `nodes on V` при наличии команды `nodes off E`.
- **10 баллов:** поддержка команды `reverse all`.
- **10 баллов:** поддержка команды `reverse E`.