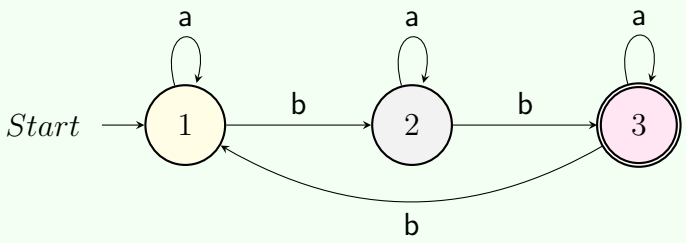
Problem

Consider the following deterministic automation:



Write a program that accepts a string of characters on a single line and, processing characters of that string left to right, determines whether that string is accepted by that deterministic automation(In which case it prints "accept") or rejected(in which case it prints "reject")

Sample input: $abab$
Expected output: $accept$
Sample input: ab
Expected output: $reject$

First, we create the transitions from the graph

```
transitions = {
    1: {'a': 1, 'b': 2},
    2: {'a': 2, 'b': 3},
    3: {'a': 3, 'b': 1},
}
```

Then we collect the terminal states

```
terminals = {3}
```

The automation in fact works with any transitions, it starts with the initial state and then traverse the graph.

```
def process(s: str, init_state, transitions: dict, terminals: set):
    state = init_state
    for c in s:
        state = transitions[state][c]
```

```
5
6       return state in terminals
```

This return $true$ or $false$, so we need an extra step to get sample outputs

```
1  print("accept" if process('abab', 1, tr, tm) else "reject")
2  print("accept" if process('ab', 1, tr, tm) else "reject")
```