Algorithm Assessment

1a

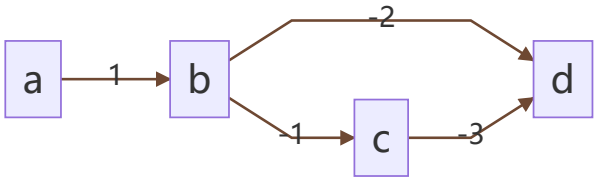
```
def loop_free(path):
    # path => [(n1, n2), (n2, n3), ...]
    seen = set()
    new_path = list()
    new_path.append(path[0])
    seen.append(path[0][1])
    for src, dst in path[1:]:
        if dst not in seen:
            new_path.append((new_path.last()[1],
dst))
        seen.add(dst)
    return new_path
```

1b

Data structure: List of tuples.

Justification: complexity is $O(n)$, we only use one iteration on the input path.

2a



2b

We want to go from `a` to `d`.

1. Pop `a` and put `b` into the queue,
$F = \{a\}, D = \{(a \to 0), (b \to 1), (c \to \inf), (d \to \inf)\}$

2. Pop `b`, and put `c` and `d` into the queue,
$F = \{a, b\}, D = \{(a \to 0), (b \to 1), (c \to 0), (d \to -1)\}$

3. Pop `d` and return.

Note that we missed the best possible path
$a \to b \to c \to d$**.**

2c

$init = \{0, \inf, \inf, \inf\}$

$relax = \{(a, b), (b, d), (b, c), (c, d)\}$

$state\ 1 = \{0, 1, \inf, \inf\}$

$relax = \{(a, c)\}$

$state\ 2 = \{0, 1, 0, \inf\}$

$relax = \{(a, d)\}$

$state\ 3 = \{0, 1, 0, -3\}$

Return -3

3

Do not know how to do.

4a