0 The Masking Tape Game

This game was developed during a slow Thursday in October, 2019, by a Computer Science group whom shall remain nameless.

1 Prerequisites

- 1 roll of masking tape
  - May be partially used to introduce variance
  - Brand *tesa*® works well
- 1 marker which works on masking tape[2]

2 Rules

1. The turn-order must be decided by the algorithm described in section A[3]
2. The roller must obey the following conditions:
   (a) Any roll must be thrown from behind the back line
   (b) Any roll must begin rolling within two meters of the back line[4]
   (c) Underhand throws are the only kind of throws allowed
   (d) A piece of tape with the name of the participant is placed on the final position
      i. The final position is not necessarily the furthest position. If the roll bounces back from an obstacle then that resulting position is the final position.
3. Participants waiting on their turn must not jeer or otherwise impact the performance of the roller
4. If a participant blocks the rolling roll, rolled by the roller, their mark is reset
5. The winner of the game is the participant who has their mark the furthest from the back line
6. The back line is marked by red masking tape near the door-frame of room 1.221 in Novi 9, SLV 312
7. Variations on these rules that introduce so-called *drinking-rules* must be submitted by PR before being considered valid[5]

[1] Contributions are welcome at `https://github.com/cogitantium/masking-tape-game`
[2] Ball-point pens do not work well
[3] If you can run the algorithm in your head, then that's fine
[4] This is to avoid yeeting the roll - instead of rolling the roll
[5] These rules need not be approved nor merged with `master` before being valid

A Turn-algorithm

The turn-order is based on the lexicographic order of the reversed SHA256-hash of each participant's name.

```
(lambda hashlib, sys: [(i == 0) and bool(print(f" #:         NAME | HASH\n
↪  {''.join(['=' for i in range(25)])}")) or bool(print(x)) for i, x in
↪  enumerate([f"{i:>2}: {x[0]:>12} | {x[1][:7]}" for i, x in enumerate(sorted([(name,
↪  bool(sha.update(name.encode("UTF-8"))) or str(sha.hexdigest())[::-1]) for name,
↪  sha in [(name, hashlib.new("sha256")) for name in sys.argv[1:]]], key=lambda x:
↪  x[1]))])])(__import__("hashlib"), __import__("sys"))
```

Naturally, this snippet is self-documenting. On the following input[6]

```
python gen.py foo bar foo_bar john hüttel rené_hüttel
```

The resulting table of turn-order is generated:

```
 #:         NAME | HASH
 =========================
 0:       hüttel | 3e18a42
 1:          bar | 9bf8fbf
 2:         john | a706cec
 3:      foo_bar | ab517ce
 4:  rené_hüttel | ca7f318
 5:          foo | ea7e662
```

To no surprise, Hüttel is the participant to start.

[6]Note that names of participants may not contain spaces. Instead, use an agreed-upon delimiter.