PACTF 2017 Tiebreaker

Background

The Hekkers (TM) have gained access to the cable which connects a datacenter to the open. After deep packet analysis, the Hekkers discover that a popular website runs its insecure HTTP website out of the compromised datacenter. The Hekkers want to recover the user's original passwords, so they can log into their accounts on other websites (a significant portion of the site's users also use the same login credentials on other sites).

However, the site did take *some* precautions. While they didn't use HTTPS, they *did* hash their passwords... sort of. When a user logs into the site, their browser sends two important pieces of information:

1. A hash of the password (implemented in `incredible_hash.py`)
2. The beginning characters of the password

All full passwords are required to be between eight and thirty-two characters long, inclusive.

The Problem

Your task is to implement a hash-reverser. Given the two pieces of data included in the login request (a string of the first few characters of the password and the full password's hash), generate and return a full password string which begins with the string included in the login request and hashes to the given hash value. The starting string may be empty, the entire password, or anything in between.

Every password may include any of the following characters: ``0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&'()*+,-./:;?@[\]^_`{|}~`` and a space (`u"\u0020"`).

The encoding is UTF-8.

If there are multiple correct answers to a test case, your program should return any one of the correct answers.

Scoring

Your team's solution will be scored based on its efficiency (speed) and accuracy. Your solution does not need to solve every test case perfectly. Find the right balance between speed and accuracy. Your team's score will be determined according to the following formula:

```
score = (1/time) * (correct/total)^2
```

where `time` is the average runtime of your solution, `correct` is the number of times your solution correctly 'reversed' the hash, and `total` is the total number of times your solution was tested. In simple english, the score is accuracy squared divided by average time.

The scoring mechanism has been included in the problem source. When assessing solutions, we will run `scorer.py` and assume that `solution.py` has properly implemented the `solution(starting_string, hash)` function.

In order to ensure that all teams' solutions are fairly tested, we will test each team's solution multiple times on a dedicated machine to minimize timing & load anomalies.

Rules

1. The solution must be implemented entirely in Python 2.7. You may not use any libraries outside of the Python 2.7 Standard Library.
2. Only one solution may be received per team. If multiple solutions are received, only the first one will be considered.
3. Solutions must be emailed to **contact@pactf.com** by **Sunday, May 21st at 12:00 PM ET**. **The email subject must be "Team Name - Tiebreaker Solution".**
4. Your solution may not interact with the filesystem or the network in any way. *We will read your code and verify this!*

You may email contact@pactf.com if any part of the problem is unclear. Happy hash breaking!