535. Encode and Decode TinyURL

TinyURL is a URL shortening service where you enter a URL such as `https://leetcode.com/problems/design-tinyurl` and it returns a short URL such as `http://tinyurl.com/4e9iAk`. Design a class to encode a URL and decode a tiny URL.

There is no restriction on how your encode/decode algorithm should work. You just need to ensure that a URL can be encoded to a tiny URL and the tiny URL can be decoded to the original URL.

Implement the `Solution` class:

- `Solution()` Initializes the object of the system.
- `String encode(String longUrl)` Returns a tiny URL for the given `longUrl`.
- `String decode(String shortUrl)` Returns the original long URL for the given `shortUrl`. It is guaranteed that the given `shortUrl` was encoded by the same object.

Example 1:

```
Input: url = "https://leetcode.com/problems/design-tinyurl"
Output: "https://leetcode.com/problems/design-tinyurl"

Explanation:
Solution obj = new Solution();
string tiny = obj.encode(url); // returns the encoded tiny url.
string ans = obj.decode(tiny); // returns the original url after deconding
it.
```

```
import random, string
class Codec:
    def __init__(self):
        self.url_pair = {}

    def encode(self, longUrl: str) -> str:
        """Encodes a URL to a shortened URL.
        """
        suffix = string.ascii_letters + string.digits
        tiny = "http://tinyurl.com/"+''.join(random.choice(suffix) for _
in range(6))
        # randomSuffix = self.generateRandomString()
        # tiny = 'http://' +'tinyurl.com' + '/' + randomSuffix
        self.url_pair[tiny] = longUrl
```

```
        # print(randomSuffix)
        return tiny

    def decode(self, shortUrl: str) -> str:
        """Decodes a shortened URL to its original URL.
        """
        return self.url_pair[shortUrl]
```

Just know about random and string libs.