Exercise 11

Task 1

b)

Problem 1: Reading files

The `preloadSounds` method is reading files from the file system. This slows down the test cases.

Problem 2: Constructor

There is way too much logic (apart from assignments) happening in the constructor of `AudioLoadHelper`. An initialization method named `preloadSounds` is called, which creates a map of the `preloadedSounds`. This depends on a global state: A folder with audio files. This makes writing test cases complicated.

Solution for both:

Because problems 1 and 2 are so intertwined we provide a solution to fix them both at once.

The map of preloaded sounds should be given as an input parameter. Similarly to slide 439 of the lecture, you can use Guice to Inject the `preloadedSounds`.

Writing a test case is way easier now. We can e.g. pass a HashMap consisting of just one mocked `SoundBridge` object.

```
@Inject
public AudioLoadHelper(HashMap<String, SoundBridge>
preloadedSounds){
    this.preloadedSounds = preloadedSounds;
    // ...
}

@Provides
HashMap<String, SoundBridge> getPreloadedSounds(){
    // logic reading from file system and returning loaded audio
files
}
```