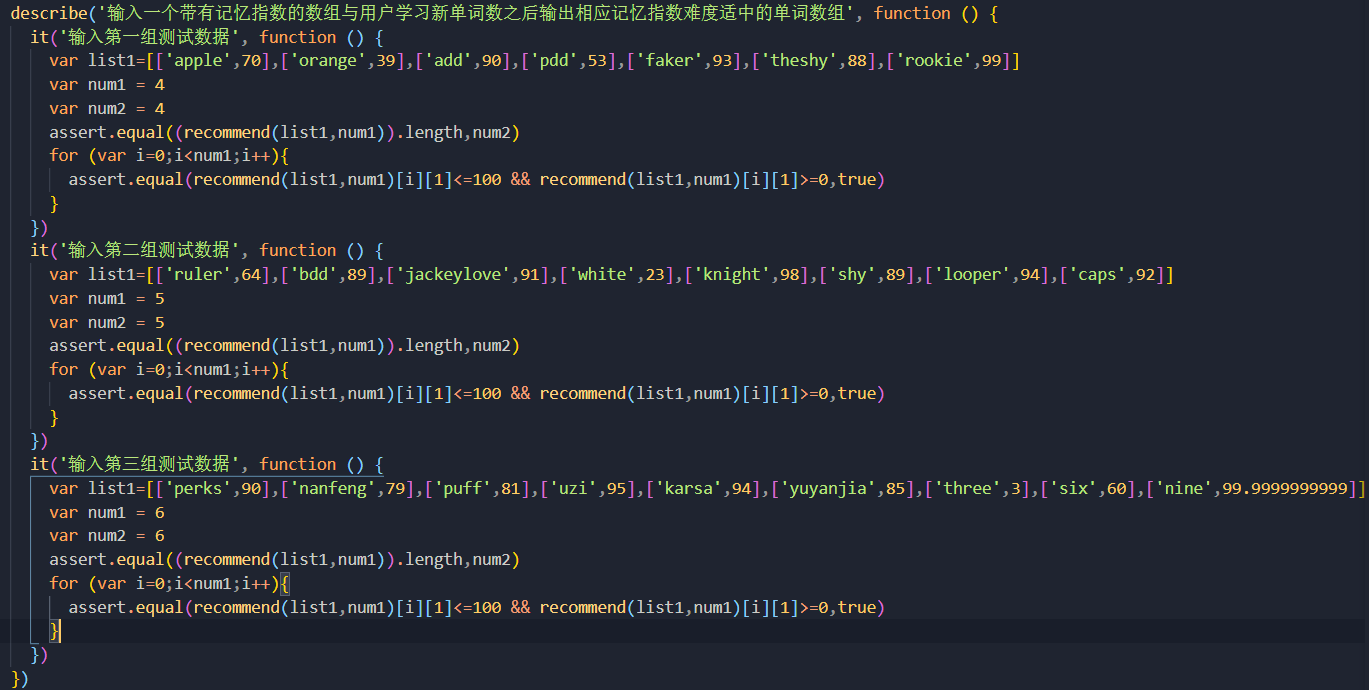
小程序单元测试总结

微信小程序本身有专门的测试工具集miniprogram-simulate支持自定义组件在nodejs 单线程运行，但由于前端工作并未有编写自定义组件的需求，我们并未使用该测试工具集。因为我们的app的核心主要是神经网络的计算以及相关的数据处理，又因为我们主体的程序都是采用JavaScript编写，所以我们决定采用mocha的框架对关键函数进行测试。

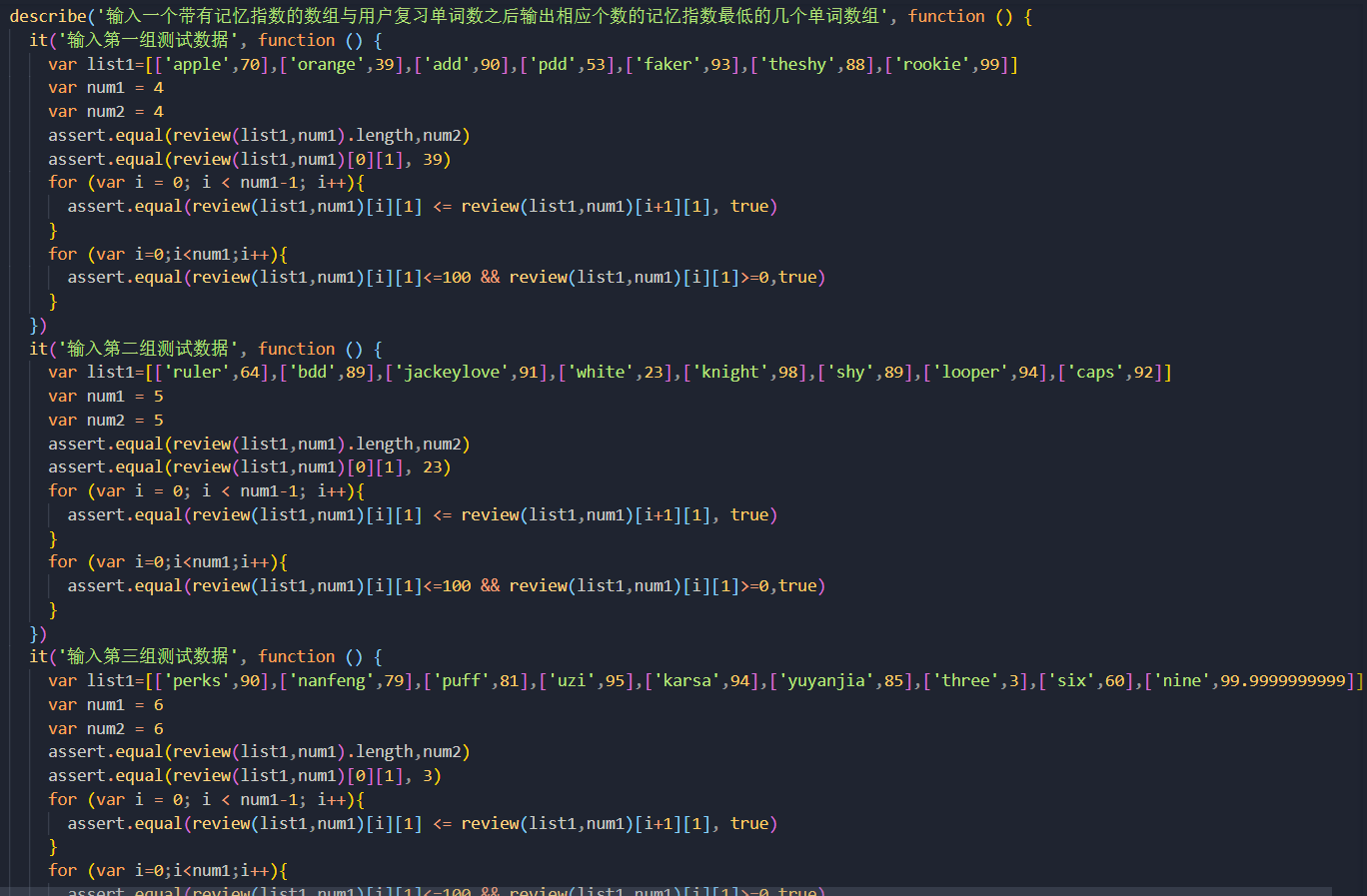
首先我们进行了遗忘函数的测试，主要的想法就是随机输入一些大于零小于一百的记忆指数，按照遗忘函数的算法得出正确的输出。我们针对各种情况都做了测试，最后都得到了正确的结果。



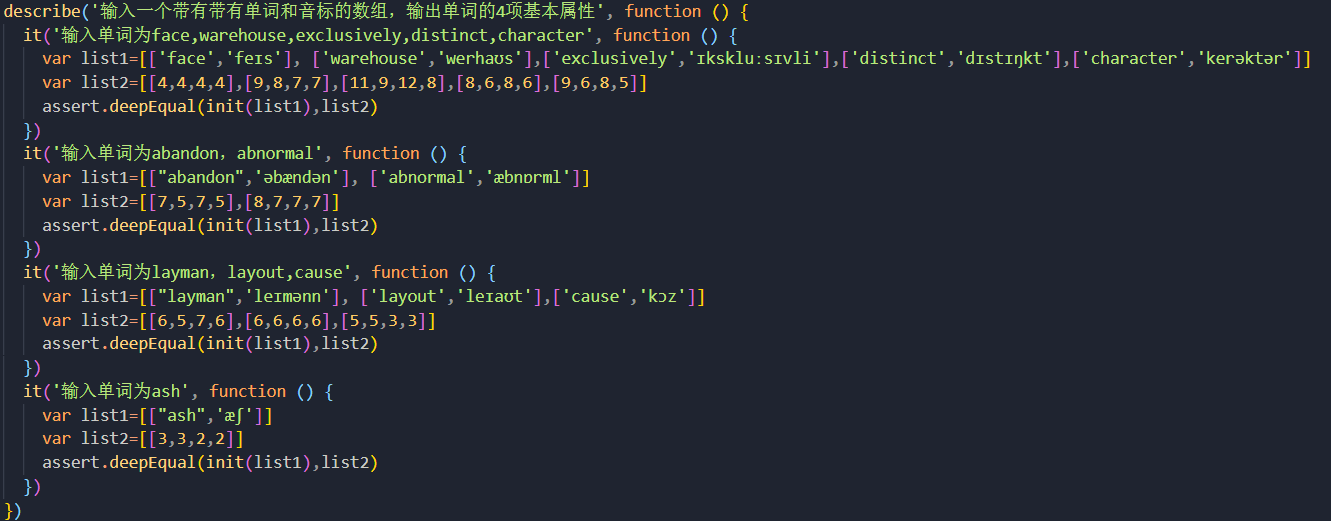
接着我们进行智能推荐函数的测试，由于我们对智能推荐的定义就是给用户推荐难度适中的新单词，但是这个难度适中并不好定义，因此主要的想法就是输入带有记忆指数的单词数组，以及用户当天想要背诵的单词数量，只要输出单词数量正确，记忆指数都是在0到100之间那么我们就算测试成功了。同样地，我们也手动做了一系列的输入输出来保证我们程序的普适性，最后发现结果也都是正确的。



再接着我们进行高效复习函数的测试，这个函数的主要目的是，输入用户背过的旧单词，以及用户当天想要复习的单词数，输出记忆指数最低的（即用户最不熟悉的）相应数量的单词。因此我们做的测试主要是判断输出的单词数目是否正确，记忆指数的范围是否正确，以及输出的单词列表记忆指数排序是否正确。同样地，我们也手动做了一系列的输入输出来保证我们程序的普适性，最后发现结果也都是正确的。



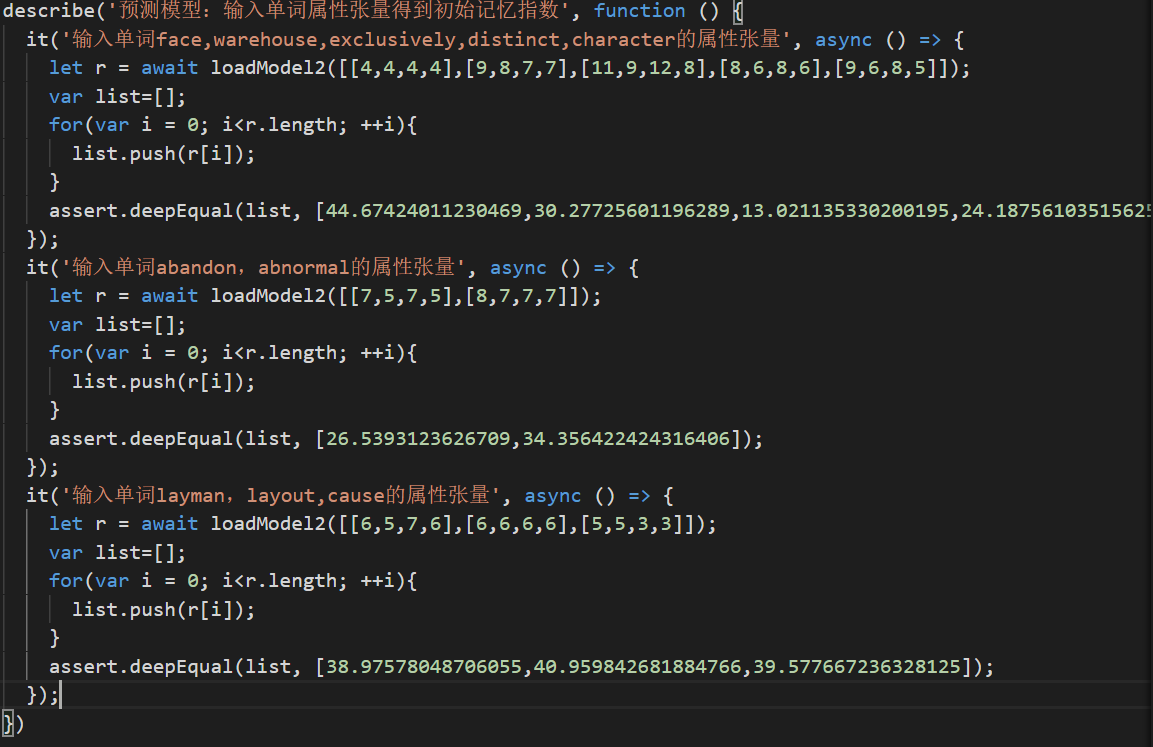
然后我们做了模型二的数据初始化函数的测试，这个函数的主要目的是输入单词的拼写以及音标，输出四个单词的基本属性，即神经网络2的输入参数。测试的方法就是手动写出输入输出，放到测试框架里运行。



最后我们做了关于神经网络两个函数的测试。这两个函数一个调用预测模型根据单词的固有属性初始化未背过单词的记忆指数，另一个则根据用户背单词过程中的各项数据对初始记忆指数进行调整。

首先模仿其他函数的测试过程编写了测试文件，发现测试结果一直报错，分析报错信息后发现每次调用神经网络模型后返回的数组始终是一个空值，说明函数并未调用成功。再次查阅mocha测试教程后发现由于两个函数中涉及到了对储存在腾讯云桶里的神经网络模型的调用，虽然async/await使代码看起来像是同步的，但它在后台是异步和非阻塞的，需要用到异步测试。按照异步测试的步骤编写好测试文件后，可以看到函数返回的值不再为空，说明调用成功。

使用assert.deepEqual判断后，神经网络模型返回结果和预测值始终不等。分析测试报错信息后发现神经网络的输出值并非javascript标准的数组类型，而是Float32Array，对此我们将输出结果转成标准数组类型后再进行比较，可以看到两个模型均能够正确处理数据。



对于以上的测试文件，我们使用[mochawesome](http://adamgruber.github.io/mochawesome/)模块处理测试结果，生成最终HTML格式的单元测试报告。