

# Python程序设计报告

设计题目：诗词问答游戏

学生姓名：陈嘉乐

专 业：计算机科学与技术

班 级：21-3班

学 号：2021218152

指导教师：马学森

完成日期：2023-11-03

# （一）需求和规格说明

**问题描述：**这是一款帮助初中学生背诗词的小程序。建立诗词库，可从诗词库中随机抽取诗词，并提供补全诗句、猜诗名、猜作者等玩法。在屏幕上显示问题和要填写的内容，用户对空格处的内容进行补全或者对问题进行回答，程序判断填补是否正确。

**编程任务：**

1. 从网页获取诗词，以字典形式存储。
2. 随机选择一首诗。
3. 判断用户选择的游戏模式，补全模式下随机抽取一句诗，再随机隐去上半句或者下半句；猜作者和猜诗名模式就是显示随机选择的那首诗，让用户填入作者名或者诗名。
4. 用户输入给出答案，程序判断用户的输入结果与答案是否匹配。

# （二）设计

## 1．设计思想

本题主要考察了解网页结构和正则表达式的相关内容，熟悉网页各标签的属性及内容，了解爬虫的机理及反爬虫规则，掌握网页数据爬取方法。我选择学习和使用requests库学习、掌握和使用.get()、.post()方法、select()、find()、find\_all()等方法。同时尝试使用了解lxml库中xpath()方法在了解搜索引擎规则后，用编程方式下载搜索数据。

在获取网页中的诗词内容后，将诗词存入字典和列表中，设计游戏规则，整个程序建立在switch菜单的大框架下，根据用户键入相应选项实现相应功能和选择的游戏模式，随机抽取字典和列表中的诗词，调用相应的游戏模式函数进行游戏和判断结果。

## 2. 设计表示

1. 类：

|  |  |  |  |
| --- | --- | --- | --- |
| **类名** | **成员类别** | **成员名** | **描述** |
| getPoem | 函数 | \_\_init\_\_ (self) | 初始化函数，放入诗词网页源地址 |
| get\_html(self) | 获取网页地址URL |
| parse(self) | 分析网页 |
| get\_poem\_list(self) | 从网页获取诗词相关内容并存入列表中 |
| 数据 | head | 诗题 |
| author | 作者 |
| content | 诗内容 |
| play\_items | 函数 | \_\_init\_\_ (self) | 初始化函数 |
| clean\_poem(self) | 清理诗词库 |
| fill\_sentence(self) | 游戏模式1：填写诗词的函数 |
| guess\_title(self) | 游戏模式2：猜诗题的函数 |
| guess\_author(self) | 游戏模式3：猜作者的函数 |
| 数据 | pattern | 要替换的内容 |
| sentence | 随机选择的诗句存入的列表数据 |
| target | 要填写的目标诗句 |
| response | 用户的输入答案 |
| title | 诗题名称 |

1. 函数：

|  |  |
| --- | --- |
| **函数名称** | **描述** |
| startGame() | 开始游戏，用户输入年级信息，显示主菜单 |

## 3. 核心算法

该程序主要有两个核心部分，一部分放在类getPoem中获取指定的诗词网页的诗词内容并存储在列表和字典中，另一部分放在类play\_items中设计游戏模式的规则和判断过程。

在类getPoem中先获得指定网页的url等信息资源，然后利用lxml库中xpath()方法用编程方式下载搜索数据。利用网页中不同内容放在不同的容器和标签选择器下，通过正则表达式合理正确获取所需要的内容。

head = title.xpath('./a/text()')[0]即可获得诗题信息

author = page.xpath('//p[@class="source"]/a[1]/text()')[0]即可获得作者信息

content=[word.strip('\n') forwordinpage.xpath('//div[@class="left"]/div[@class="sons"][''1]/div[@class="cont"]/div[''@class="contson"]//text()')]即可获得诗词的文本内容。

再将他们放入字典数据类型中，利用字典中键值对对应的特性，可方便查找一首诗的诗题、作者和诗歌内容三个部分：sector.append({'title': head, 'author': author, 'content': content})

在类play\_items中制定游戏模式和规则，判断用户回答正确与否。诗句补全函数fill\_sentence（self）中，先随机获得某句诗后，利用split（）函数，以逗号“，”为分割点返回得到包含上下句的列表，列表的第一个元素即为上句，而第二个元素即为下句，向用户展示其中一句的信息，匹配用户答案是否与另一句相同，相同即为回答正确并给出提示，不同即为错误，给出提示同时再给出答案。猜诗题函数guess\_title(self) 与猜作者函数guess\_author(self)的逻辑基本相似，在随机到一首诗后，通过字典给出内容，匹配用户输入的答案是否与该诗的诗名或者作者对应即可，利用字典的特性即可快速查找。

最后在函数startGame（）使用两层while循环，第一层while循环等待用户输入年级信息，利用if语句判断即可；在第二层while循环中展示游戏菜单，等待用户输入选择相应的游戏模式，同样利用if语句判断并选择调用相应的函数实现功能即可。

# （三）用户手册

程序运行后首先显示正在初始化诗歌信息，当屏幕上显示初始完成字样后会提示用户输入年级信息，用户可输入的年级信息有：七上、七下、八上、八下、九上、九下。

在年级信息输入完成后会让用户选择游戏模式，根据用户键入的数字会有不同功能。

键入数字1，选择长句补全游戏模式，用户根据一句诗的上句，输入并补全下句。

键入数字2，选择猜诗名游戏模式，用户根据屏幕上显示的部分诗的内容，输入猜测的诗名（不需要加书名号）。

键入数字3，选择猜作者游戏模式，用户根据屏幕上显示的部分诗的内容，输入猜测的作者名。

键入数字4，选择退出游戏模式选择环节，重新输入年级信息。

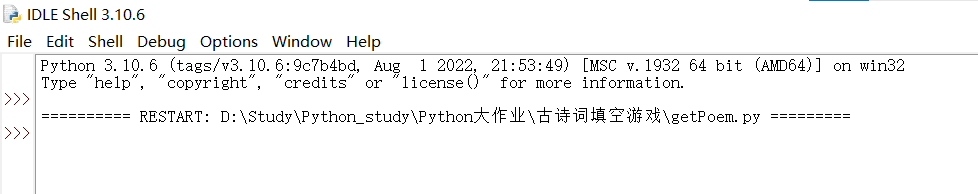
键入数字0，选择退出游戏，结束程序。

在游戏模式1、2、3中，若回答正确，则提示回答正确；若回答错误，则提示用户回答错误，并给出正确答案在屏幕上。

每一次游戏模式结束后都会让用户重新选择游戏模式。

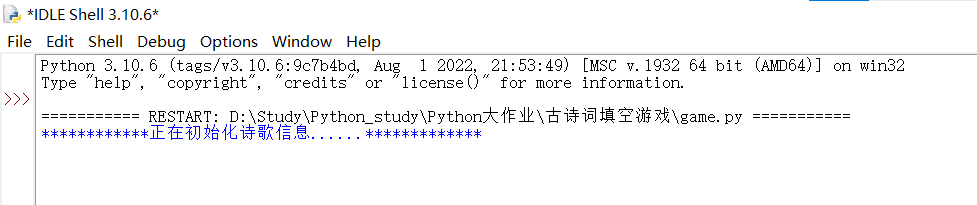
# （四）调试及测试

## 1. 构建诗词库：

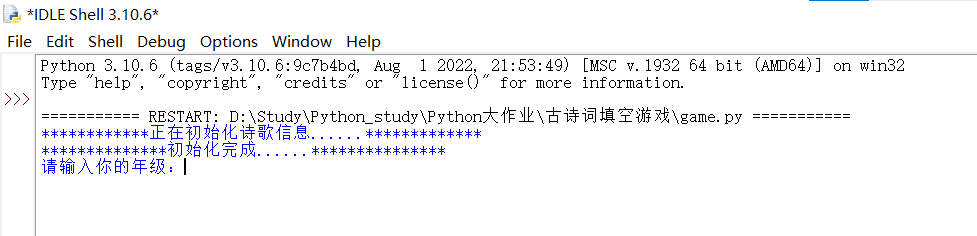


**从网页获取资源，构建诗词库**

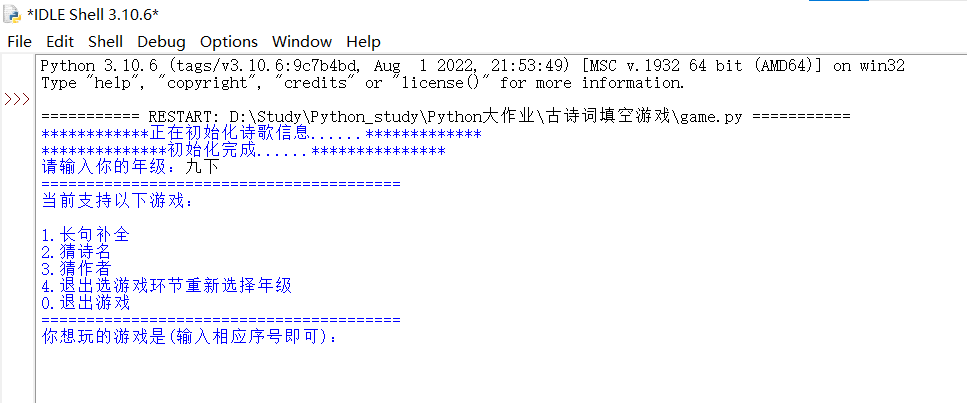
## 2. 游戏测试：



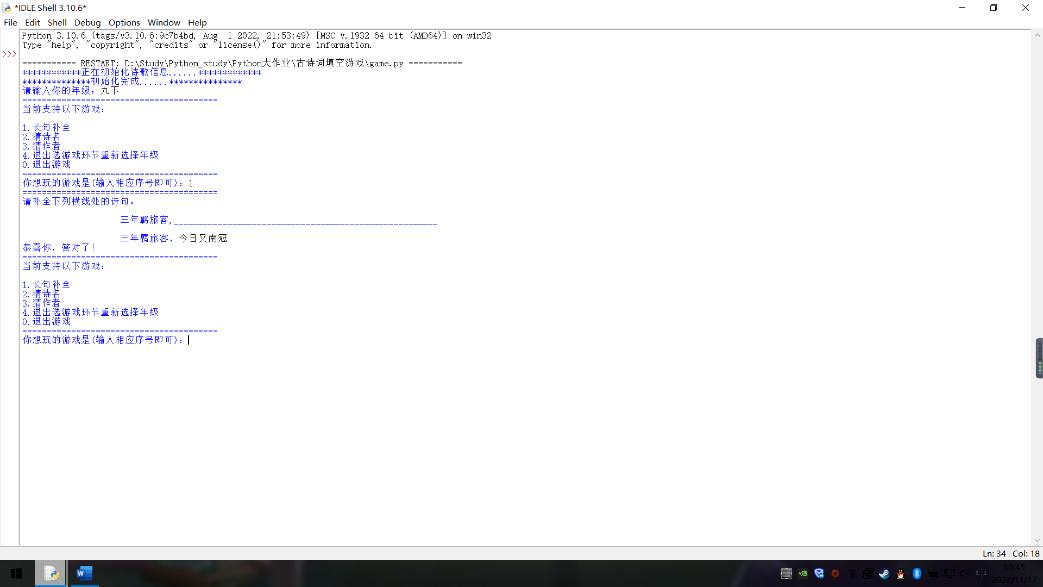
**正在初始化诗歌信息**



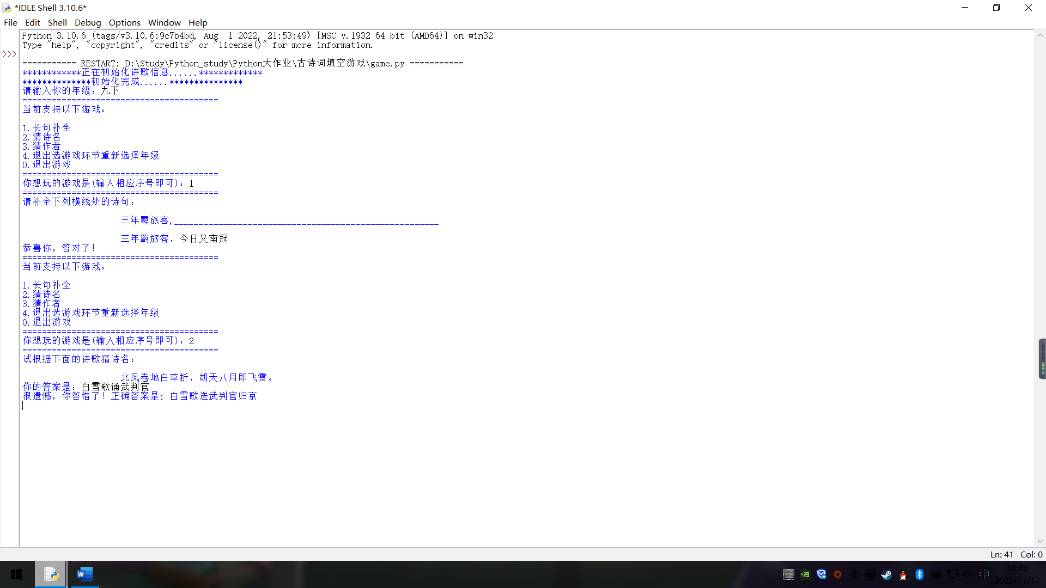
**用户输入年级信息**



**游戏模式菜单**



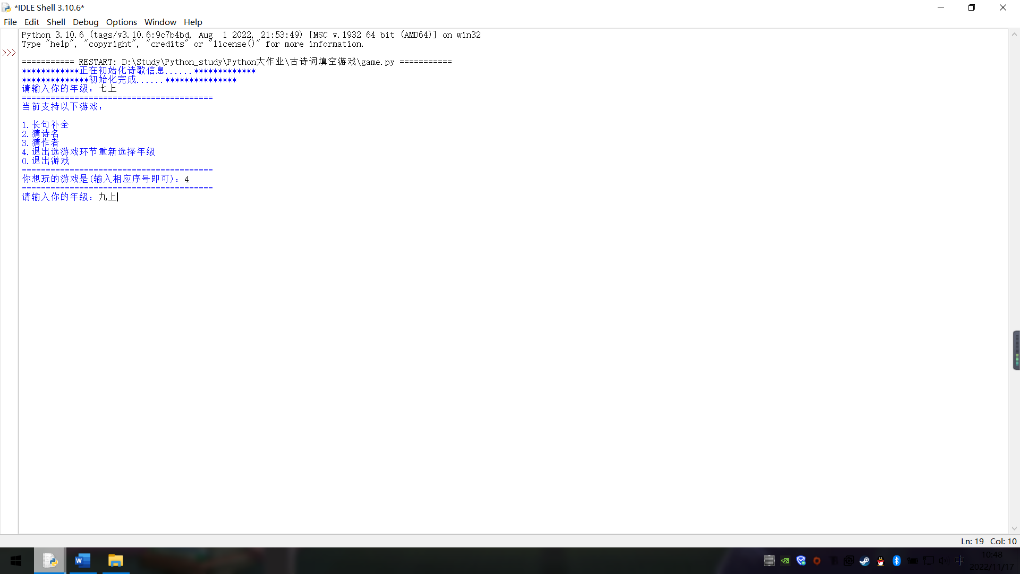
**键入数字1，选择长句补全**



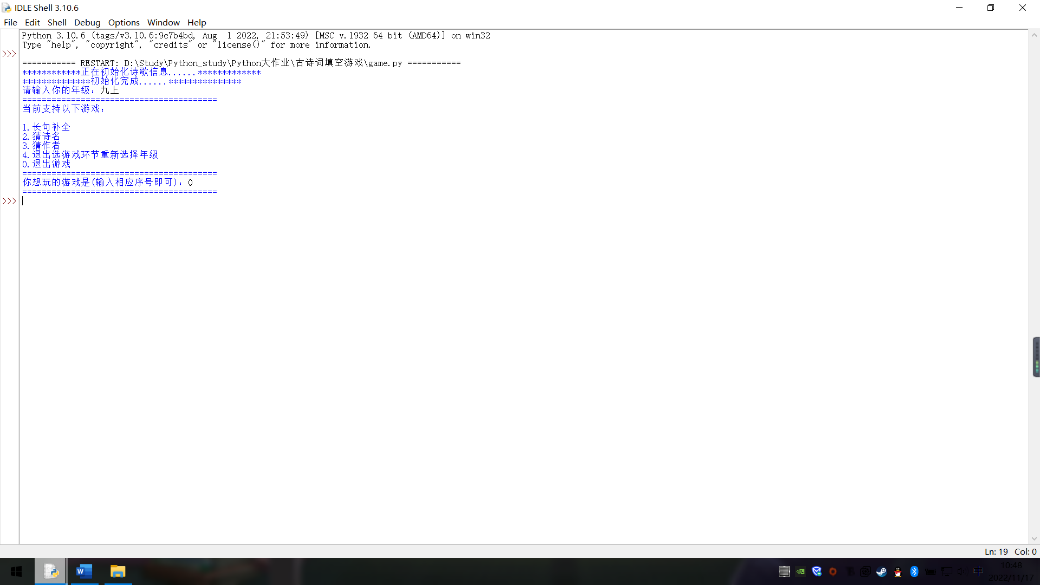
**键入数字2，选择猜诗名**



**键入数字3，选择猜作者**



**键入数字4，选择重新输入年级信息**



**键入数字5，选择退出游戏**

## 3. 不足之处：

（1）没有采用数据库进行存储，因此每次初始化慢（需要先从网上抓取），且对网站造成一定的影响

（2）目前这个程序只能先通过getPoem.py提前从指定的网站后去资源并建立好一个诗词库。

（3）由于词的字数不对称，以‘。’切割，必然存在问题。

（4）由于原网站中在包含诗词内容的<p></p>内还含有<br>标签，所以在爬取资源存入列表中时会导致某些列表内容为空，在游戏过程中会导致输出同样为空，没有诗句内容。在网上搜索解决办法后曾尝试在爬取过程中去除<br>标签，也曾尝试先转成字符串形式，然后去掉\n，但都效果不理想，至今也没有好办法。

# (五) 感想

这道题对现在的我来说还是挺难的，尤其是从网站上获取诗词内容这一技术能力在我设计出这道题之前还没有学习和掌握。但是在经过书本阅读、课外视频学习和利用网络查看借鉴他人的代码后算是首次独立完成了本题目。

我做这道题前是先列了一个清单，把要实现的功能罗列出来，比如获取网页资源，功能选择，隐去诗句，判断正误，再判断哪些实现某些功能的代码是可以结合在一起、一起实现的，最后组合起来。最开始写的时候也没有考虑时间复杂度，只盼着能实现能运行就好，写完之后才发现有些地方可以优化，原先部分代码写的繁琐冗杂，而后经过算法优化才有些改进。同时在测试程序时出现了很多不便于用户输入，或者忘记了应该输入什么样的格式内容等问题，也让我明白以后要尽量在写程序之前先考虑好用户会怎么使用这个程序，从使用者的角度出发。分开做时都比较顺利的，但是合起来就出现了很多小问题，比如参数传递出错，变量命名不统一等等。我觉得在亲身解决这些问题后，这些经历都会对我以后的程序设计有所帮助。

对正则表达式实现复杂匹配深入学习了一下后，有了更深的感悟，同时也对xpath解析浅浅地学习了以下，更重要的是对python获取网页资源内容的学习使我对python的强大功能有了更深刻的认识。尤其在字符串上的处理，python相对于c语言也是给程序员极大的便利。而python不仅如此它的库也很多，正因为它强大的库，让编程变得不再艰难。但是，我认为python虽然在许多方面相对于c语言比较方便，但也有其相对于弱一点的方面，比如说for循环等方面。虽然一学期下来，我对python的学习也仅仅只是它的基础方面，但python的强大，也是足足地吸引着我，希望自己能够在不断地学习中，将python学习的更加好。

# 附录：

#getPoem.py

import requests

from lxml import etree

class getPoem:

def \_\_init\_\_(self):

self.url = 'https://so.gushiwen.cn/gushi/chuzhong.aspx'

self.headers = {'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, '

'like Gecko) Chrome/84.0.4147.135 Safari/537.36'}

self.pl = dict()

def get\_html(self):

try:

response = requests.get(url=self.url, headers=self.headers)

if response.status\_code == 200:

return response.text

else:

print('页面获取错误：状态码{}'.format(response.status\_code))

except Exception as e:

raise ConnectionError('获取网页失败！')

def parse(self):

response = self.get\_html()

source = etree.HTML(response)

chapters = source.xpath('//div[@class="typecont"]')

return chapters

def get\_poem\_list(self):

for chapter in self.parse():

grade = chapter.xpath('./div/strong/text()')[0]

# print(grade)

titles = chapter.xpath('./span')

sector = list()

for title in titles:

head = title.xpath('./a/text()')[0]

# print(head)

poem\_link = title.xpath('./a/@href')[0]

# print(poem\_link)

poem = requests.get(poem\_link)

page = etree.HTML(poem.text)

author = page.xpath('//p[@class="source"]/a[1]/text()')[0]

content = [word.strip('\n') for word in page.xpath('//div[@class="left"]/div[@class="sons"]['

'1]/div[@class="cont"]/div['

'@class="contson"]//text()')]

sector.append({'title': head, 'author': author, 'content': content})

self.pl.update({grade: sector})

# play\_items.py

import random

import re

class play\_items:

def \_\_init\_\_(self, poem):

self.poem = poem

self.clean\_poem()

def clean\_poem(self):

pattern = re.compile(r'(\([^)]\*\))')

self.poem['content'] = pattern.sub('', self.poem['content'][0])

def fill\_sentence(self):

content = self.poem['content']

# 删除注解

content = re.sub(r'（', '(', content)

content = re.sub(r'）', ')', content)

pattern = re.compile(r'(\([^)]\*\))')

content = [words for words in pattern.sub('', content).split('。') if words != '']

sentence = random.choices(content)

target = sentence[0].split('，')[0]

print('请补全下列横线处的诗句：')

print('{},{}'.format(target, '\_'\*(len(target)\*2)))

response = input('{}，'.format(target))

if response == sentence[0].split('，')[1]:

print('恭喜你，答对了！')

else:

print('很遗憾，你答错了！正确答案是: {}。该句出自{}'.format(sentence[0].split('，')[1], self.poem['title']))

def guess\_title(self):

content = self.poem['content']

title = self.poem['title']

print('试根据下面的诗歌猜诗名：')

print(''.join(content))

response = input('你的答案是：')

if response == title:

print('恭喜你，答对了！')

else:

print('很遗憾，你答错了！正确答案是: {}'.format(title))

def guess\_author(self):

content = self.poem['content']

author = self.poem['author']

print('试根据下面的诗歌猜作者：')

print(''.join(content))

response = input('你的答案是：')

if response == author:

print('恭喜你，答对了！')

else:

print('很遗憾，你答错了！作者是: {}'.format(author))

#game.py

from getPoem import getPoem

from play\_items import play\_items

import random

import time

def startGame():

# 1. 获取诗歌

print("{:\*^40}".format('正在初始化诗歌信息......'))

instance\_api = getPoem()

get\_poem = instance\_api.get\_poem\_list()

poem\_list = instance\_api.pl

print("{:\*^40}".format('初始化完成......'))

active = True

while active:

player\_grade = input('请输入你的年级：')

poem\_valid = []

if player\_grade == '七上':

poem\_valid = poem\_list['七年级上册'] + poem\_list['七年级上册(课外古诗词诵读)']

elif player\_grade == '七下':

poem\_valid = poem\_list['七年级下册'] + poem\_list['七年级下册(课外古诗词诵读)']

elif player\_grade == '八上':

poem\_valid = poem\_list['八年级上册'] + poem\_list['八年级上册(课外古诗词诵读)']

elif player\_grade == '八下':

poem\_valid = poem\_list['八年级下册'] + poem\_list['八年级下册(课外古诗词诵读)']

elif player\_grade == '九上':

poem\_valid = poem\_list['九年级上册'] + poem\_list['九年级上册(课外古诗词诵读)']

elif player\_grade == '九下':

poem\_valid = poem\_list['九年级下册'] + poem\_list['九年级下册(课外古诗词诵读)']

else:

print('请重新选择年级：')

# 将诗歌打乱

random.shuffle(poem\_valid)

if poem\_valid:

while True:

print('{}'.format('='\*40))

print('当前支持以下游戏：\n')

print('1.长句补全')

print('2.猜诗名')

print('3.猜作者')

print('4.退出选游戏环节重新选择年级')

print('0.退出游戏')

print('{}'.format('='\*40))

game\_type = eval(input('你想玩的游戏是(输入相应序号即可)：'))

print('{}'.format('='\*40))

# 随机获取一首诗歌

poem = random.choices(poem\_valid)[0]

# print(poem)

# 初始化游戏

game = play\_items(poem)

# 选择游戏类型

if game\_type == 0:

active = False

break

if game\_type == 1:

game.fill\_sentence()

time.sleep(3)

if game\_type == 2:

game.guess\_title()

time.sleep(3)

if game\_type == 3:

game.guess\_author()

time.sleep(3)

if game\_type == 4:

break

startGame()