سری تمرین شماره یک درس یادگیری عمیق



علی رجبی نکو

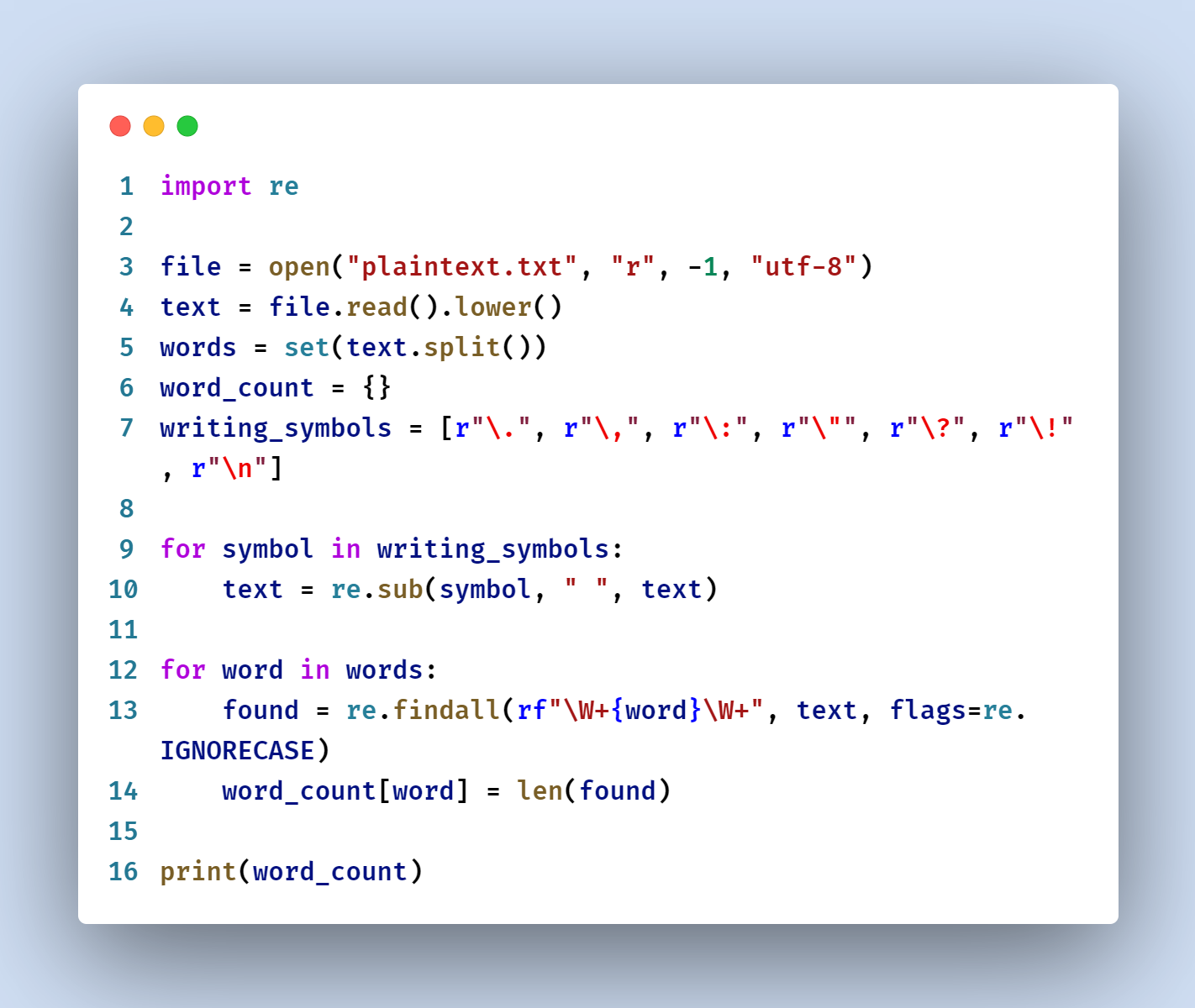
شماره دانشجویی: 402532007

دانشجوی کارشناسی ارشد مهندسی نرم افزار دانشگاه باهنر کرمان

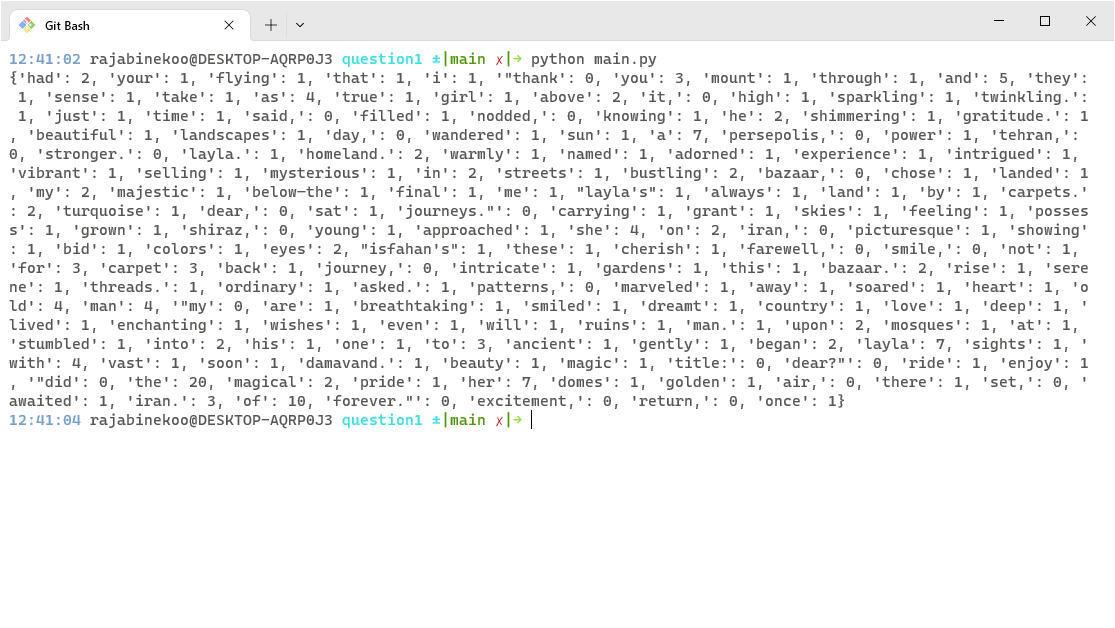
استاد: خانم دکتر قاسمیان

لینک کدها: <https://github.com/ali-rajabinekoo/deepLearning>

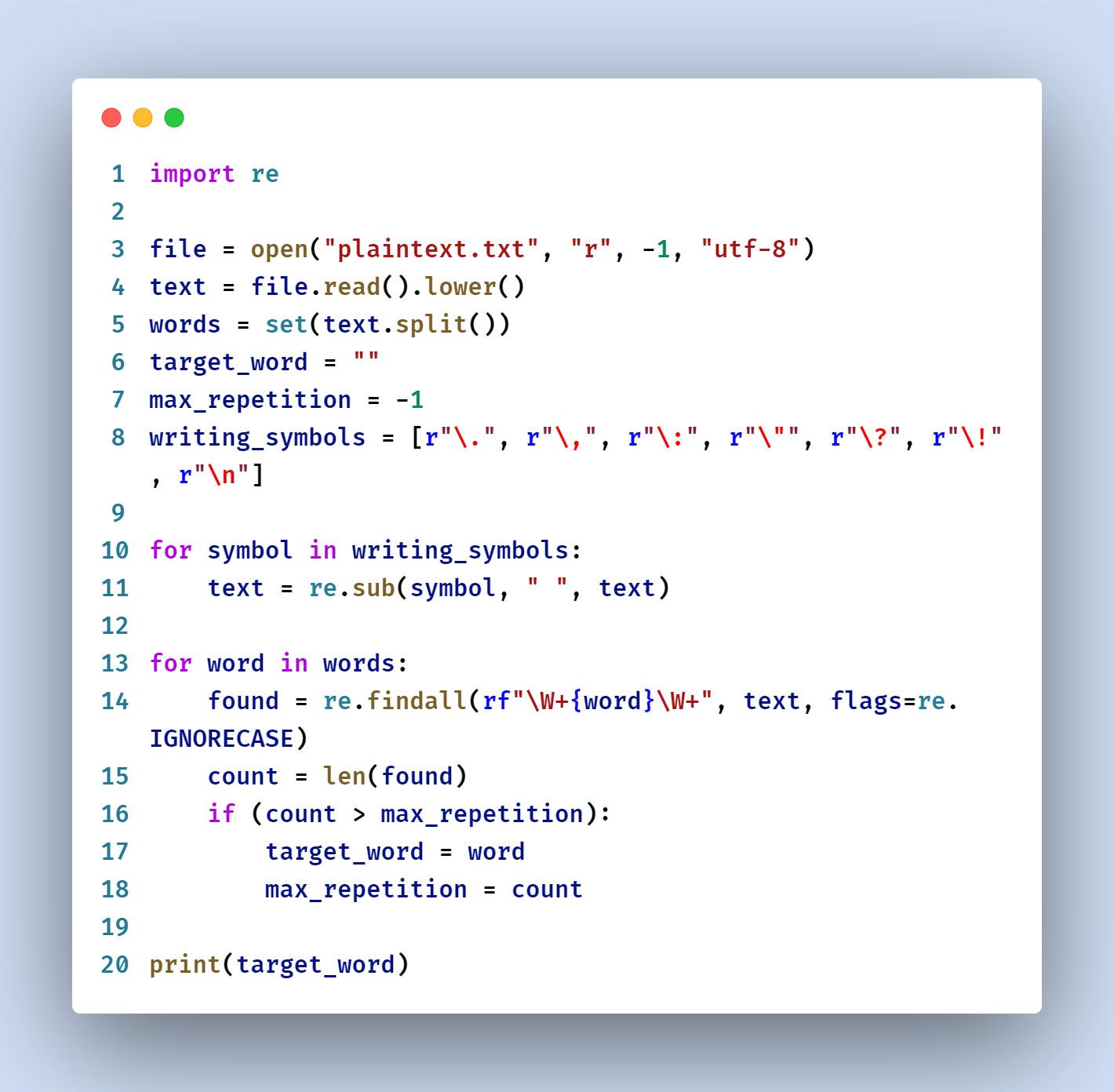
تمرین اول:



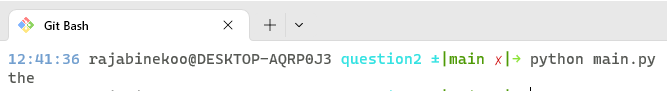
1. در خط ابتدایی ماژول re که ماژولی برای توابع regular expression است استفاده شده است.
2. دو خط بعدی، فابل plaintext.txt را با تابع open و مد read ، به صورت utf8 میخواند زیرا کاراکتری هایی مثل خط فاصله دارد. متن را به صورت lowercase نیز تبدیل میکنیم تا regex زدن ساده تر شود.
3. در خط بعدی کلمات متن خوانده شده را استخراج میکند و داخل لیست میریزد و سپس آنرا تبدیل به set یا مجموعه میکند زیرا آیتم های تکراری داخل set وجود نخواهد داشت و حلقه کوتاه تر به پایان میرسد.
4. در خط بعدی دیکشنری خالی ایجاد میکنیم که داخل ان شماره تعداد کلمات را ذخیره کنیم.
5. در خطر بعدی نماد های نگارشی که موجب تداخل عملکرد regex شود را داخل ارایه قرار میدهیم تا در خطر بعدی هریک از این نماد ها را داخل متن حذف کنیم. با تابع re.sub این کار شدنی است که با space جایگزین میکنیم زیرا داخل رشته regex به space ابتدا و انتهای کلمات نیاز داریم.
6. سپس تمامی کلمات استخراج شده را با تابع re.findall که روشی است برای دریافت لیست تمامی رشته هایی که با رشته regex مطابقت دارد، دریافت میکنیم. در خط بعدی داخل دیکشنری مورد نظر، بصورت کلید و مقدار، کلمه و تعداد دفعات تکرار را نگهداری میکنیم.



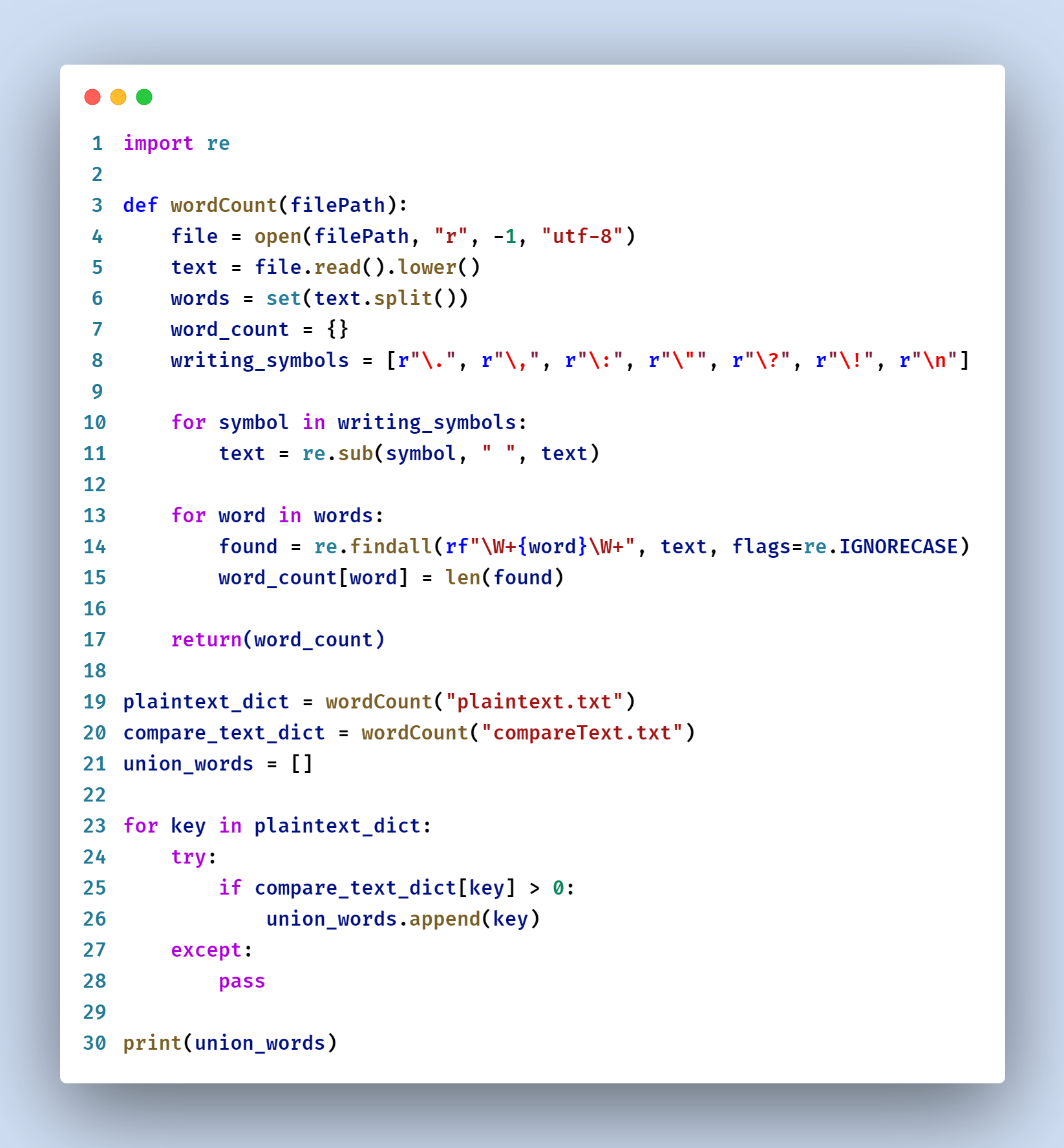
تمرین دوم:



1. در خط ابتدایی ماژول re که ماژولی برای توابع regular expression است استفاده شده است.
2. دو خط بعدی، فابل plaintext.txt را با تابع open و مد read ، به صورت utf8 میخواند زیرا کاراکتری هایی مثل خط فاصله دارد. متن را به صورت lowercase نیز تبدیل میکنیم تا regex زدن ساده تر شود.
3. در خط بعدی کلمات متن خوانده شده را استخراج میکند و داخل لیست میریزد و سپس آنرا تبدیل به set یا مجموعه میکند زیرا آیتم های تکراری داخل set وجود نخواهد داشت و حلقه کوتاه تر به پایان میرسد.
4. در دو خط بعدی، اولا رشته خالی ایجاد میکنیم که داخل آن کلمه با بیشترین تکرار قرار میگیرد، دوما متغیری ایجاد میکنیم که بیشترین میزان تکرار کلمه مورد نظر را داخل خود ذخیره میکند و البته مقدار پیشفرض آن منفی یک است زیرا گستره عددی مورد نظر برای طول کلمات حسابی است و -1 کوچک از کوچکترین عدد داخل این مجموعه است (یعنی صفر).
5. در خطر بعدی نماد های نگارشی که موجب تداخل عملکرد regex شود را داخل ارایه قرار میدهیم تا در خطر بعدی هریک از این نماد ها را داخل متن حذف کنیم. با تابع re.sub این کار شدنی است که با space جایگزین میکنیم زیرا داخل رشته regex به space ابتدا و انتهای کلمات نیاز داریم.
6. سپس تمامی کلمات استخراج شده را با تابع re.findall که روشی است برای دریافت لیست تمامی رشته هایی که با رشته regex مطابقت دارد، دریافت میکنیم. در خط بعدی چک میکنیم که آیا طول کلمه مورد نظر بیشتر از کلمه ذخیره شده قبلی است یا خیر که اگر اینطور باشد، کلمه و طول آن را داخل دو متغیر target\_word و max-repitition قرار میدهیم. در نهایت target\_word برابر کلمه ای است که بیشترین تکرار را دارد.



تمرین سوم:



1. خط 1 تا 17 همان کد تمرین 1 است که به صورت تابع تعریف میشود و در خط 4 متغیر filePath مسیر فایل مورد نظر است که از ورودی دریافت شده است زیرا قرار است دو فایل متنی با یکدیگر مقایسه گردند و خروجی تابع دیکشنری تعداد کلمات استخراج شده است.
2. در خط 19 و 20 دو فایل مورد نظر که قرار است مقایسه گردند را به تابع wordCount میدهیم تا دیکشنری تعداد کلمات برگردد و داخل متغیر های plaintext\_dict و compare\_text\_dict قرار گیرد.
3. سپس لیستی از کلمات مشترک را به صورت خالی تعریف میکنیم که آماده ذخیره داده داخل آن باشیم. سپس حلقه ای داخل plaintext\_dict میزنیم که هر کلمه بعنوان کلید دریافت کنیم و آن کلید را نیز داخل compare\_text\_dict قرار میدهیم تا ببینیم که آیا عددی بیشتر از 0 میدهد یا خیر. بنده از جایی که برنامه نویسی زبان JS و C# انجام داده ام خروجی یا عدد میشود یا null یا undefined ولی در python خطایی مبتنی بر پیدا نکردن آیتم مورد نظر بعنوان کلید، خروجی میدهد. به همین علت از exception handling استفاده میکنیم که اگر ارور دریافت نکردیم مطمین شویم که آن کلمه به عنوان کلید در compare\_text\_dict موجود است. سپس آرایه کلمات مشترک را چاپ میکنیم.

