تحلیل‌گر لغوی

اعضای تیم:

کیمیا میرمقتدایی ۴۰۱۲۳۶۳۰۹۱

ارشیا شفیعی ۴۰۰۳۶۲۳۰۱۹

استاد پروژه:

دکتر آرش شفیعی

فهرست مطالب

|  |  |  |
| --- | --- | --- |
|  | شرح پروژه | |
|  | رویکرد کلی | |
|  | بررسی یک به یک توابع | |
|  | 1. تابع get\_tokens |  |
|  | منابع | |

شرح پروژه:

در این پروژه قصد داریم برای یک زبان برنامه‌نویسی مشابه زبان C یک کامپایلر طراحی و پیاده‌سازی کنیم. یک کامپایلر از تحلیل‌گر لغوی، تحلیل‌گر نحوی، تحلیل‌گر معنایی، تولید‌کننده و بهنیه‌ساز کد میانی، تولیدکننده و و بهینه‌شاز کد اسمبلی تشکیل شده است. در این مرحله قصد داریم تا تحلیل‌گر لغوی زبان تعریف شده را طراحی و پیاده‎سازی کنیم. به این منظور از زبان پایتون بهره برده‌ایم. همچنین از هیچ کتاب‌خانه‌ی فرعی برای این منظور استفاده نشده است.

رویکرد کلی:

رویکرد کلی در این پروژه تعریف توابعی برای بررسی نوع هر توکن (با توجه به انواع توکن‌های تعریف شده در صورت پروژه) تعریف کرده‌ایم. هربار یک خط از برنامه‌ی نوشته شده را می‌خوانیم و در آن کاراکتر به کاراکتر جلو می‌رویم و این توابع را به ترتیب داخل یک حلقه تا پایان برنامه فراخوانی می‌کنیم. همه‌ی این توابع –علاوه‌بر ویژگی‌های لازم برای هر توکن- یک مقدار True یا False برمی‌گردانند که نشان می‌دهد عبارت فرستاده شده از نوع توکن مربوطه هست یا خیر.

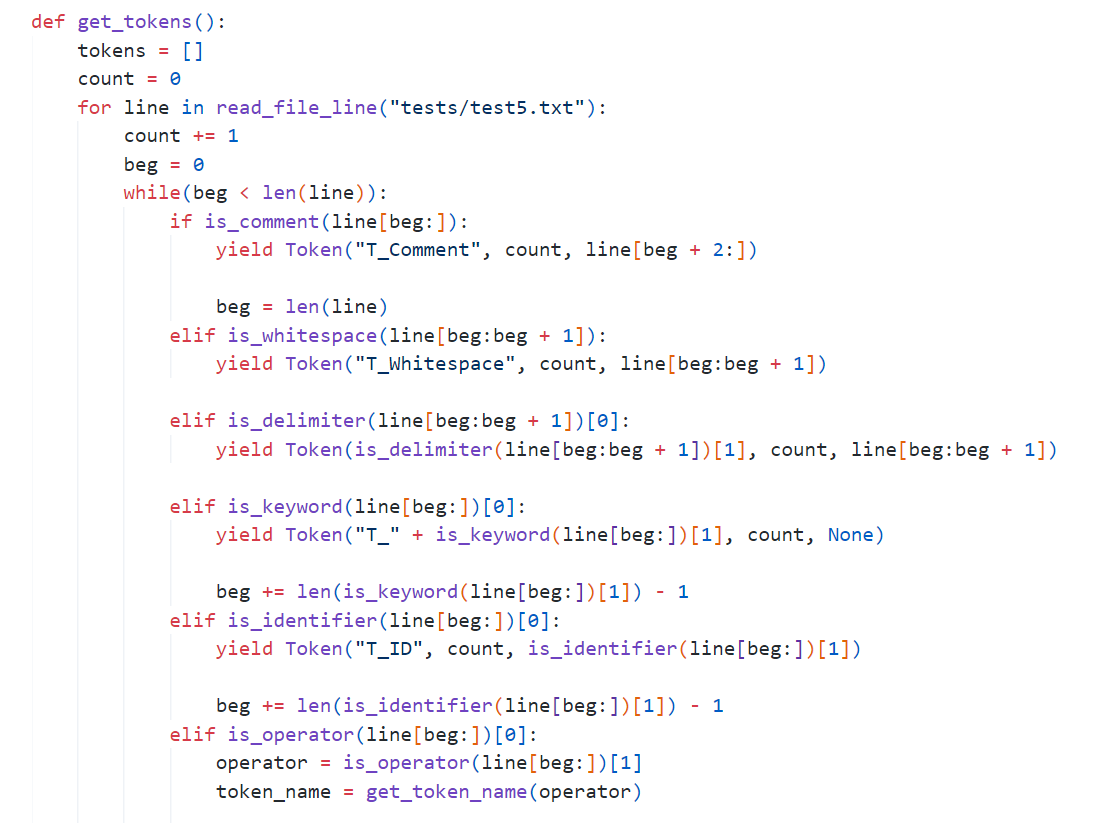
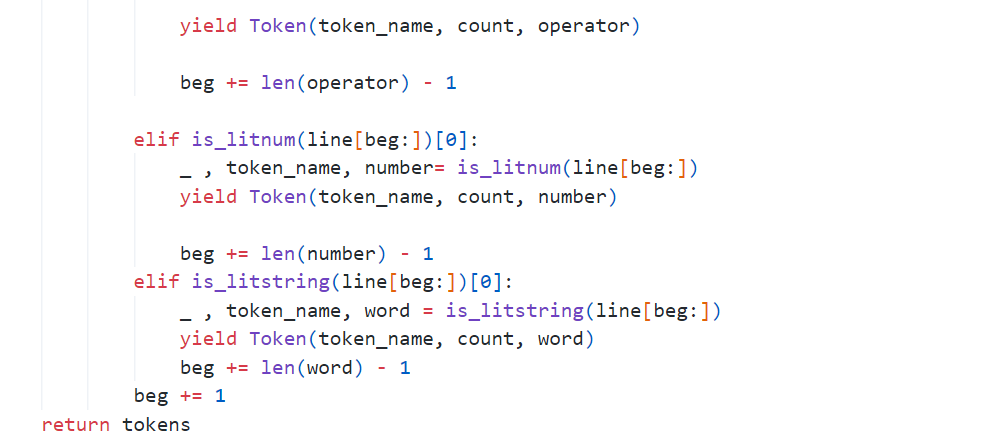
هربار مقدار یک تابع False برگرداند، تابع بعدی در دنباله‌ی توابع اجرا خواهد شد. در صورت True بودن یک تابع، توکن ما از نوع مربوط به آن تابع تشخیص داده می‌شود و بنابراین توکن آن را می‌سازیم و در خروجی قرار می‌دهیم.

بررسی یک به یک توابع:

1. تابع get\_tokens:

در این تابع به ازای هر خط داخل کد، متغیر beg را تعریف می‌کنیم که در عمل اشاره‌گری به کاراکتری از خط است که هنوز به توکنی تعلق ندارد. در هر مرحله تا پایان هر خط، beg را یکی یکی زیاد می‌کنیم تا به پایان خط برسیم. هربار با شروع از beg تا پایان خط را به یکی از توابع تشخیص‌دهنده‌ی توکن‌ها می‌دهیم تا در صورت امکان توکن درست را استخراج کنیم.

در صورتی که با شروع از beg توکنی پیدا نشود، مقدار آن را یکی زیاد می‌کنیم و دوباره امتحان می‌کنیم.

شکل 1- کد تابع get tokens