

درس: طراحی کامپایلر

عنوان: پروژه تحیل گر لغوی

عضو اول: علیرضا کریمی

شماره دانشجویی عضو اول: 993623035

عضو دوم: علی پورقیصری

شماره دانشجویی عضو دوم: 993613014

استاد: دکتر آرش شفیعی

فهرست مطالب

[1- مقدمه 2](#_Toc163339067)

[2- مجموعه متغیر های سراسری 3](#_Toc163339068)

[3- کلاس Token 3](#_Toc163339069)

[4- توابع مورد استفاده در طول کد 3](#_Toc163339071)

[1-4- is\_comment(line) 4](#_Toc163339072)

[2-4-space\_separator(line) 4](#_Toc163339073)

[3-4-space\_operator(line) 4](#_Toc163339074)

[4-4- find\_word\_occurrences(word, line) 4](#_Toc163339075)

[4-5- analyze\_line(line, row) 4](#_Toc163339076)

[4-6- analyze\_file(file\_path) 4](#_Toc163339077)

[4-7- main() 4](#_Toc163339078)

[مراجع 4](#_Toc163339079)

1- مقدمه

در فاز اول پروژه، قرار بر این است که یک تحلیلگر لغوی را از صفر با استفاده از زبان پایتون پیاده سازی کنیم. تحلیلگر لغوی در این کد با استفاده از ترکیبی از دیکشنری ها برای تعریف کلمات کلیدی، عملگرها، جداکننده ها، نقل قول ها و کاراکترهای فضای خالی پیاده سازی شده است. کلاس تعریف شده در این کد ویژگی های هر توکن را در بر می گیرد و روش های درون آن بر اساس قوانین از پیش تعریف شده، نوع توکن را تعیین می کنند. تحلیلگر خطوط ورودی را با تقسیم کردن آنها به کلمات، و سپس تکرار روی هر کلمه برای تعیین نوع رمز آن، نشانه گذاری می کند. فضاهایی در اطراف جداکننده‌ها و عملگرها برای تسهیل توکن‌سازی درج شده‌اند. به طور کلی، تحلیلگر لغوی کد خام را پردازش می‌کند، هر عنصر را به انواع نشانه‌های خاص دسته‌بندی می‌کند و نمایشی ساختاریافته از کد را برای پردازش یا تحلیل بیشتر ارائه می‌دهد.

2- مجموعه متغیر های سراسری

این متغیرهای سراسری در سرتاسر کد برای دسته‌بندی نشانه‌ها در طول تحلیل لغوی استفاده می‌شوند. با استفاده از آنها منطق متمرکز برای تعریف و مدیریت انواع توکن ها، افزایش خوانایی، قابلیت نگهداری و انعطاف پذیری کد را میتوان فارهم نمود. این کاراکتر ها شامل انواع علائم منطقی، نشانه گذاری، بلوک بندی و ... می باشد که معمولا در سایر زبان ها نیز وجود دارند.

3- کلاس Token

کلاس Token در این کد به عنوان ساختاری برای نمایش توکن های فردی در کد استفاده شده است. این کلاس شامل ویژگی‌هایی برای ذخیره واژگان، نوع آن، نام نشانه یا شناسه، و موقعیت آن در کد (ردیف و ستون) است. علاوه بر این، شامل روش هایی برای تعیین نوع نشانه بر اساس معیارهای از پیش تعریف شده مانند کلمات کلیدی، عملگرها، جداکننده ها، نقل قول ها، فضای خالی، شناسه ها، ثابت ها و نظرات می باشد. متد **analyze\_lexeme()** توکن را بر اساس این معیارها دسته بندی می کند و مقادیر مناسب نوع و نام توکن را به آن اختصاص می دهد. به طور کلی، کلاس **Token** منطق تجزیه و تحلیل و طبقه‌بندی نشانه‌ها را در فرآیند تحلیل واژگانی یک زبان برنامه‌نویسی در بر می‌گیرد.

4- توابع مورد استفاده در طول کد

در طول برنامه یکسری توابع استفاده شده است که پروسه تحلیل لغوی از طریق این توابع انجام می‌گیرد. این توابع به شرح زیر می‌باشند:

1-4- is\_comment(line): این تابع بررسی می‌کند که آیا یک خط کد معین یک کامنت است یا نه، با بررسی اینکه آیا با نشانگر کامنت // شروع می‌شود یا خیر.

2-4-space\_separator(line)**:** این تابع فضاهای اطراف جداکننده ها (مانند پرانتز، پرانتز، براکت، نقطه ویرگول و کاما) را در یک خط کد مشخص درج می کند. این مرحله برای توکن سازی صحیح کد ضروری است.

3-4-space\_operator(line)**:** این تابع مانند قبل، فضاهای اطراف عملگرها (مانند عملگرهای حسابی، مقایسه و منطقی) را در یک خط کد معین درج می کند. این تضمین می کند که هر اپراتور در طول توکنایز شدن به عنوان یک توکن جداگانه در نظر گرفته می شود.

4-4- find\_word\_occurrences(word, line): این تابع با در نظر گرفتن هر کاراکتر رزرو شده، همه موارد تکرار یک کلمه را در یک خط کد پیدا می کند. لیستی از موقعیت هایی را که کلمه در خط ظاهر می شود، برمی گرداند.

4-5- analyze\_line(line, row): این تابع یک خط کد را تجزیه و تحلیل می کند، آن را نشانه گذاری می کند و لیستی از اشیاء Token را که نشان دهنده هر نشانه یافت شده در خط است، برمی گرداند. از توابع قبلی برای پیش پردازش خط، پیش از توکن سازی استفاده می کند.

4-6- analyze\_file(file\_path): این تابع یک فایل را خط به خط می خواند، از کامنت ها می گذرد و هر خط بدون نظر را با استفاده از تابع analyze\_line(file\_path) آنالیز می کند. جزئیات هر نشانه موجود در فایل را چاپ می کند، مانند واژگان، نوع نشانه و موقعیت در کد. این تابع به عنوان نقطه ورودی برای توکن کردن کل فایل عمل می کند.

4-7- main(): این تابع جاییست که analyze\_file(file\_path) فراخوانی می‌شود.

مراجع

* chat.openai.com
* [Introduction of Lexical Analysis - GeeksforGeeks](https://www.geeksforgeeks.org/introduction-of-lexical-analysis/)
* <https://www.youtube.com/watch?v=O4Bt_CyZWbI>
* https://medium.com/@pythonmembers.club/building-a-lexer-in-python-a-tutorial-3b6de161fe84