**گزارش فاز دوم پروژه درس طراحی کامپایلر:**

**PLY:**

PLY یک ابزار مناسب برای پیاده سازی بخش مهم کامپایلر lex و yacc به زبان پایتون می باشد. PLY شامل دو ماژول مجزا lex.py و yacc.py است که هردو در پکیج ply موجود هستند. در این فاز ما میخواهیم از ماژول lex.py استفاده نماییم و در ادامه نحوه عملکرد این ماژول را شرح میدهیم.

**LEX:**

Lex.py در شکستن برنامه ورودی به مجموعه توکن های تعریف شده به صورت عبارت منظم مورد اسفاده قرار میگیرد. به این صورت که کاراکتر های برنامه ورودی را بررسی می نماید و با توکن های معرفی شده به صورت عبارت منظم مقایسه مینماید در صورتی که کاراکتر ها به گونه ای چیده شده و با یکی از توکن ها مطابقت داشته باشند آنگاه آ« دسته کاراکتر را از برنامه ورودی جدا نموده و در گروه توکن مورد نظر قرار داده و به صورت خوروجی برمی گرداند و سپس به سراغ کاراکتر های بعدی رفته و اینگونه تمام برنامه ورودی را به صورت توکن های تعریف شده شسکته و تقسیم بندی مینماید.

**Token List:**

همانگونه که در بالا اشاره نمودیم، ابزار Lex شامل یک بخش ضروری به نام token list می باشد که شامل لیست تمام توکن هایی است که ممکن است در برنامه شناسایی شوند.

**Specification of tokens:**

با اعلام لیست توکن ها می بایست هر توکن را جداگانه با استفاده از قوانین عبارت های منظم تعریف نماییم که این عمل توسط ماژول re در پایتون انجام می شود. این شناسایی به دو صورت انجام میشود: اگر توکن مورد نظر ساده باشد کافیست تعریف قانون مربوط به آن توکن به صورت یک string انجام شود. در غیر اینصورت اگر نیاز به انجام عملیات های مختلف در حین تعریف توکن باشد باید تعریف توکن مورد نظر را به صورت یک تابع پیاده سازی کنیم.

**Line numbers and positional information:**

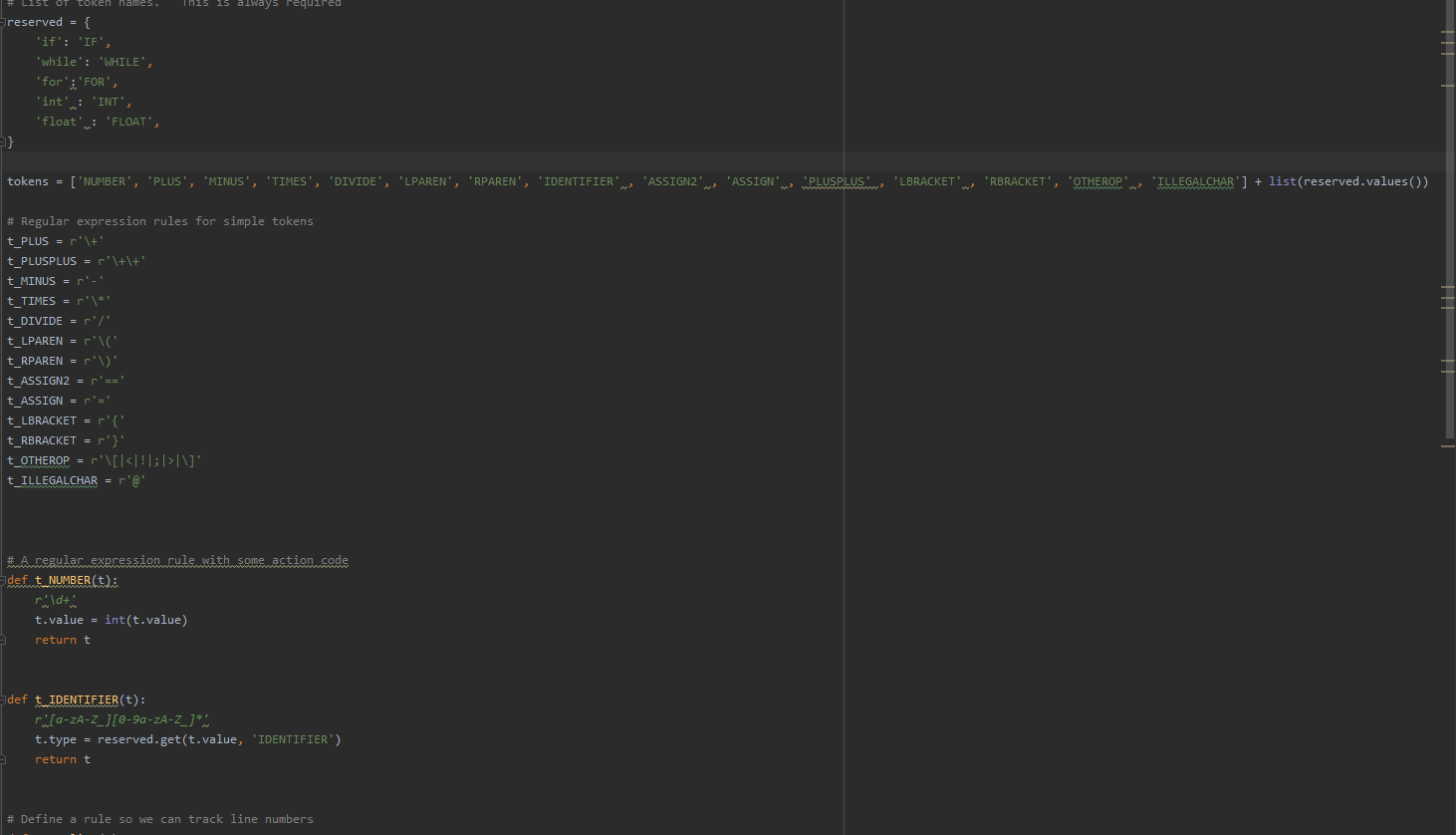
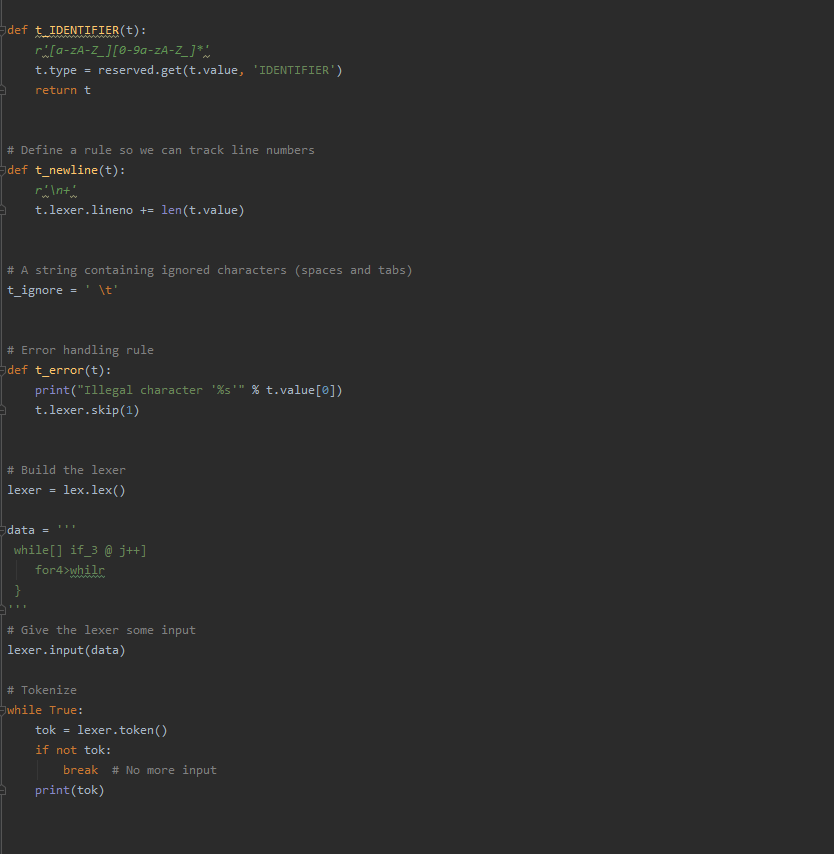
به صورت پیش فرض ابزار lex.py توانایی شمارش خط را ندارد برای این عمل تابعی تحت عنوان t\_newline میسازیم که لکسر با خواندن کاراکتر \n توکن newline را شناسایی نموده و به متغیر lineno یک واحد میافزاید. این تابع مقاداری را بازنمیگرداند.

**Ignored characters:**

قانون t\_ignore برای lex.py رزرو شده است تا کاراکتر هایی که باید در برنامه ورودی نادیده گرفته شوند را مورد بررسی قرار ندهد مانند فاصله ها و tabها

**Building and using the lexer:**

در نهایت برای استفاده از یک لکسر و مشخص کردن توکن های فایل ورودی ابتدا با استفاده از تابع lex.lex() یک لکسر ایجاد میکنیم مثل: Lexer = lex.lex(). پس از ایجاد لکسر مورد نظر میتوانیم از دو متد lexer.input(data) برای دادن فایل برای مشخص نمودن توکن های آن و همچنین lexer.token() برای بازگرداندن توکن یافت شده از فایل ورودی، استفاده نماییم.



**Building and using the lexer**

**Line numbers and positional information**

**Specification of tokens:**

Functions for Complex Tokens

**Ignored characters**

**Specification of tokens:**

Functions for Complex Tokens

**Specification of tokens:**

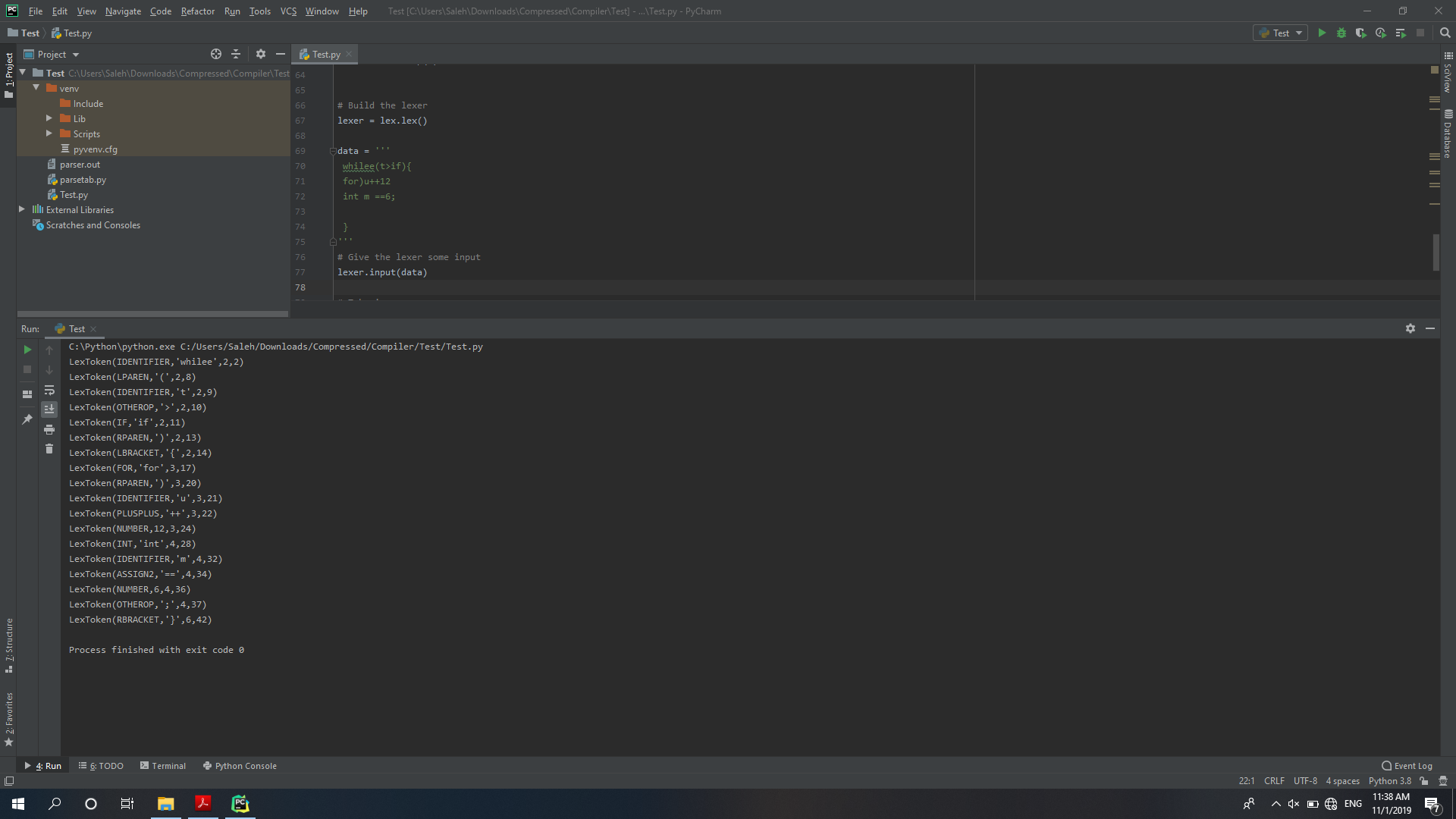
Regular Expressions for Simple Tokens

**Specification of tokens:**

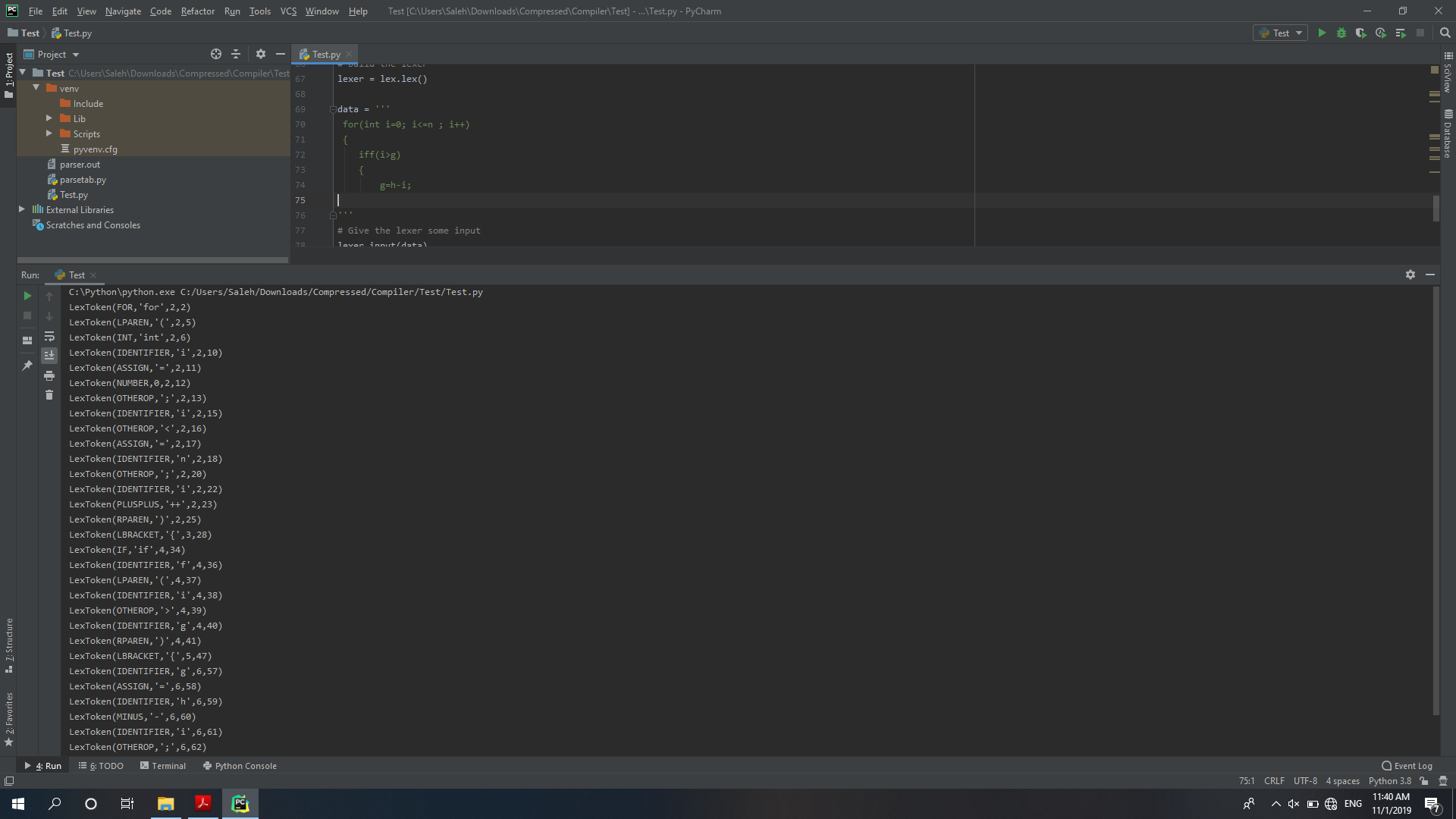
Reserved List for KEYWORDS

**Token List**

برنامه های تست شده به همراه خروجی آن ها:



برنامه های تست شده به همراه خروجی آن ها:



برنامه های تست شده به همراه خروجی آن ها:

