به نام خدا

فاز اول پروژه درس اصول طراحی کامپایلرها

مقدمه

همان طور که در طول ترم با مفاهیم و تکنیک های آن آشنا می شوید، برای پیاده سازی کامپایلرها از ابزارهایی استفاده می شود. برای تشخیص کلمات و علائم (lexeme) از نرم افزار lexer استفاده می شود. کلمات دریافت‌شده از این ابزار، در ادامه در اختیار ابزار yacc قرار می گیرد. این ابزار با توجه به گرامر زبانی که به عنوان ورودی دریافت می‌کند، یک parse tree ایجاد می‌کند که به برنامه نویس این امکان را می‌دهد که برای هر دستور از گرامر، تعدادی فعالیت[1] تعریف کند تا بتواند به وسیله آن کد میانی تولید کند. با این ابزار، در فازهای بعدی پروژه بیشتر آشنا خواهید شد. در این فاز، با نحوه کار ابزار lexer و همچنین، مفاهیم اولیه طراحی کامپایلر از قبیل Regular Definition آشنا خواهید شد.

فاز اول پروژه

در این فاز از پروژه، با توجه به گرامر زبانی که در اختیار شما قرار گرفته است باید ورودی ابزار lexer را آماده کنید. Synatx ورودی ابزار، همان طور که در کلاس تدریسیار معرفی شد، در فایل ضمیمه نیز آمده است. به طور خلاصه داریم:

قالب فایل lexer، برای استفاده از ابزار JFlex که در آن به زبان جاوا می‌توان کد نوشت، به ۳ قسمت که با %% از هم جدا می شوند، تقسیم می شود. قسمت اول user code، قسمت دوم declaration و قسمت سوم rules خواهد بود.

در قسمت دوم با

```
% {
// your java code
%}
```

می توانید کد جاوا بنویسید.

syntax قسمت declaration چنین است:

```
Digit = [0-9]
```

[1] action

Letter=[a-z]

Id={letter}({letter}|{digit})*

برای اشاره به متغیرهایی که در این بخش تعریف شده‌اند، مثل digit باید از {} استفاده کنید، مثل بالا.

برای regular definitionها داریم:

* یعنی عبارت قبل از آن می‌تواند بین صفر تا هر تعدادی تکرار شود.

+ یعنی عبارت قبل از آن می‌تواند بین یک تا هر تعدادی تکرار شود.

؟ یعنی عبارت قبل از آن می‌تواند صفر یا یک بار تکرار شود.

| همان مفهوم OR را دارد.

با استفاده از مفاهیم بالا می توان یک الگو برای تشخیص مفاهیم مختلف ارائه داد. برای مثال اگر بخواهیم به شکل ساده الگوی اعداد حقیقی را تعریف کنیم، می‌توانیم بنویسیم:

realNum = $\{digit\}^{+}([.]\{digit\}^{+})?$

در مورد بالا توان، leading zero و trailing zero رعایت نشده است که می‌توانید به عنوان تمرین آن را انجام دهید.

به طور کلی از regular definition برای مشخص کردن الگوی کلمات کلیدی، constantها، ";"، ID و ... استفاده می شود. ابزار lexer با استفاده از تعاریفی که ارائه می‌دهید، یک آتاماتا برای تشخیص نوع هر lexeme استفاده می کند. هر بار یک کاراکتر از ورودی می‌خواند و با توجه به آتاماتای موجود از یک state به state دیگر می‌رود تا در نهایت نوع آن lexeme را تشخیص دهد. توضیحات کامل در مورد نحوه کار lexer در جزوه موجود است.

در قسمت آخر که ruleها مطرح می‌شوند، داریم:

“Key_word” or {variable} {Java code}

قواعد کامل در مورد lexer در یک tutorial در کنار lexer برای شما ارسال شده است.

تکالیف فاز اول

- ابتدا با توجه به گرامر داده‌شده، یک کد نمونه بنویسید که شامل همه قواعد گرامر باشد. هر چه این برنامه نمونه کاملتر باشد، بهتر است. (خودتان نیز می‌توانید در فازهای بعدی، از این برنامه به عنوان مورد آزمون استفاده کنید.)
- با توجه به گرامر داده شده، regular definition مناسب و کلمات کلیدی را بیابید و با توجه به آن فایل ورودی lexer را آماده کنید.
- فایل آماده‌شده را به وسیله ابزار lex اجرا کرده، فایل جاوا یا C تولیدشده را کامپایل کنید و خروجی تولید شده را در قالب یک فایل txt. ارسال کنید.

برای خروجی می‌توانید از هر الگویی استفاده کنید. برای نمونه می‌توانید از الگوی زیر استفاده کنید:

- If IF-keyword -
- A ID symbol table address
- 123.2 real number -

پیاده سازی symbol table در این فاز اختیاری است. در ادامه، این مباحث در کلاس تدریس خواهند شد که در فازهای بعدی آن را پیاده‌سازی خواهید کرد.

- برای جمع‌بندی، فایل‌هایی که باید از طریق سایت درس، آپلود کنید شامل فایل کد نمونه، ورودی lexer، کد تولید شده توسط lexer و فایل خروجی است. همه این فایل ها به جز فایل C یا جاوای تولید شده توسط lexer را باید به صورت پرینت‌شده به استاد نیز تحویل دهید.

برای هر گونه سوال درباره پروژه، می‌توانید با farzin.heidary@gmail.com تماس بگیرید.