به نام خدا

فاز سوم پروژه درس اصول طراحی کامپایلر

۱- تولید جدول نمادها

یکی از بخش‌هایی که باید به خروجی قسمت دوم پروژه اضافه گردد، اضافه شدن جدول نمادها می‌باشد. هدف از تولید این جدول در این پروژه ذخیره‌سازی متغیرهای تعریف شده در متن برنامه می‌باشد. نحوه پیاده‌سازی این بخش برعهده دانشجو می‌باشد اما به عنوان یک راه پیشنهادی برای ساخت این جدول می‌توانید در فایل yacc یک ساختمان داده به صورت global تعریف کنید و پس از کشف یک تعریف متغیر در تحلیل نحوی در قسمت action فایل yacc که در این فایل توضیح داده شده، به صورت یک function call متغیر را به ساختمان داده اضافه نمایید.

۲- تولید کد میانی

هدف از انجام این بخش تولید کد میانی برای برنامه نوشته شده مطابق گرامر می‌باشد. همانطور که برای انجام بخش دوم پروژه خروجی lex تغییر یافت و تا به ورودی قسمت دوم تبدیل شود، خروجی قسمت دوم نیز باید کمی تغییر یابد و در خروجی خود یک درخت نحوی انتزاعی تولید کند. برای ساخت این نوع خروجی به قسمت parse tree این لینک مراجعه کنید. بعد از تولید درخت نحوی انتزاعی برای تولید کد میانی باید روی این درخت پیمایش DFS انجام دهید و روی آن کد سه آدرسه تولید نمایید.

برای ساده کردن این بخش فرض شده‌است برنامه ورودی شما از ساختمان داده استفاده نمی‌کند و به صورت یک all_expression (قاعده ۲۷ گرامر) به ورودی داده می‌شود. خروجی نهایی شما کد سه آدرسه خواهد بود.

در بخش دوم پروژه، تجزیه کننده بارها lexer را فراخوانی می کند تا توکن بعدی را دریافت کند، و در هنگام تجزیه نحوی یک ورودی، یک درخت نحوی انتزاعی صریحا ایجاد می شود. پس از ساخت این درخت با اضافه کردن کد مربوط به پیمایش DFS درخت کد سه آدرسه تولید خواهد شد:

```
main() {
do{yyparse{;()
while(!feof(yyin())
 */code for traversing (dfs) the syntax tree and generating three-address code/*
}
```

پس از اضافه کردن این کد، به ازای یک ورودی دوباره روال ساخت را اجرا کنید، اگر خطایی در فاز تحلیل لغوی (lexer.lex) کشف نشود، و خطای گرامری در تحلیل نحوی (parser.y) رخ ندهد، خروجی شما کد سه آدرسه خواهد بود.

برای گرفتن ایده برای انجام این بخش می‌توانید از فایل‌های فولدر complete sample موجود در فایل پروژه استفاده نمایید.