بسمه تعالی

Parsa Yahooie
610395157

PA3 - PARSER

هدف این فاز از پروژه، مشخص کردن قوانینی به منظور ساخت یکساختمان داده درختی بود.
ابتدا به علت استفاده از ساختار union، باید برای گره‌های غیر برگ درخت، نام‌هایی مشخص کنیم. این اتفاق در خط‌های ۱۳۳ الی ۱۴۸ افتاده است. سعی شده موارد مربوط به هم در کنار هم قرار گیرند تا خواناتر شوند.
پس از آن نیاز است که اولویت عملگرها مشخص شود که این کار در خطوط ۱۵۱ الی ۱۵۹، بر اساس داده‌های بخش ۱۱ کتابچه cool-manual صورت گرفته است. توجه شود که کلمه left به معنای اولویت یک عملگر هنگام مواجهه با عملگرهای شبیه خود است که یعنی مثلا دو ضرب، از چپ شروع به انجام میشوند.
همچنین این لیست از پایین به بالا از بیشترین اولویت به کمترین اولویت میرود.
در نهایت دستور زبان را نوشتم و سعی کردم حتی الامکان سعی کردم منطبق بر ساختار تعریف شده در تعاریف بخش union باشد و همچنین کلاس‌های تعریف شده در خود بایسون برای ساخت درخت که در بخش ۶.۵ کتابچه cool-tour توضیح داده شده و ساختار هر یک از توابع داخل فایل cool-tree.cc نوشته شده بود.
در نهایت به منظور خواناتر شدن کد و جلوگیری از برخی اشتباهات و باگ‌ها، ساختار let و case را از دستور زبان expression جدا کردم.
بخش expression از روی لیست موجود در بخش ۱۰ کتابچه cool-manual الگوبرداری شده و منطبق بر همان نوشته شده.

توضیحات بخش‌های مختلف گرامر:
بخش program: این بخش ریشه درخت است و کل برنامه را در بر میگیرد. از آنجا که زبان کول مرکب چند کلاس هست، پس این بخش لیستی از کلاس‌ها را دارد.
بخش class_list_semicolon: این بخش دو حالت دارد که یکی حالت پایانی است که فقط شامل کلاس میشود و دیگری شامل لیستی از کلاس‌ها. نام این بخش به این خاطر به این شکل نوشته شده است که عبارت ’;‘ class_list_semicolon class خوانایی بیشتری داشته باشد :))))
بخش class: ساختار کلی یک کلاس را به نمایش میگذارد با حالات ممکن. اگر هیچ یک از موارد صدق نکرد، در بخش آخر به عنوان ارور شناخته خواهد شد.
بخش feature_list_semicolon: مانند لیست کلاس‌ها است.
بخش feature: توابع و متغیرها را دربر میگیرد. در صورت عدم انطباق ارور تشخیص داده میشود.
بخش formal_list_comma: مانند لیست کلاس‌هاست.
بخش formal: بیانگر ورودی‌های یک تابع است.
بخش‌های expression_list_comma و expression_list_semicolon: مانند بخش لیست کلاس‌ها هستند.
بخش expression: بیانگر تمام حالات نهایی است و باقی قابلیت‌ها را به نمایش میگذارد. بدنه let و case نیز در ادامه این بخش نوشته شده‌اند.

فایل‌های دیگر:

فایل bad.cl دارای مواردی هست که پارسر باید به خطاهای آن حساس باید و همچنین good.cl حاوی مواردی هست که سعی شده کل زبان را پوشش دهد و کاملا درست باشد.

فایل good-result.txt خروجی فایل good.cl است. خروجی bad.cl نیز به صورت اسکرین شات داخل فولدر bad-results وجود دارد.