بسم اللّه الرّحمن الرّحیم

دانشگاه صنعتی اصفهان

طراحی کامپایلر

گزارش پروژه

محمد کسائی(۹۷۳۳۳۱۳) ، نامی نذیری(۹۷۳۶۲۲۳)

۲۹ تیر ۱۴۰۰

۱ Lexical analyzer

در این بخش لازم است برنامه ورودی که به صورت رشته و دنباله‌ای از کاراکترهاست بررسی شود. به این‌منظور برای هر یک از مفاهیم موجود در زبان برنامه‌نویسی مورد نظرمان باید یک عبارت منظم طراحی و ارائه دهیم. در فایل l. تمام علائم و رشته‌های مجاز که در مجموع یک معنای واحد را تشکیل می‌دهند. این بخش‌ها را توکن می‌نامیم و از این پس کامپایلر با توکن‌ها کار می‌کند.

۲ Syntax analyzer

در فاز بعدی لازم است برنامه‌ی وارد شده را از لحاظ گرامری بررسی کنیم و چیدمان توکن‌های مختلف را نسبت به هم بررسی می‌شوند و گرامری برای توصیف کل زبان برنامه‌نویسی ارائه می‌شود.
گرامر به صورت کامل در فایل y. آمده است.
ایده‌ی کلی گرامر به این‌صورت است که برنامه شامل تعدادی تابع و تعریف متغییر جهانی Global variable است و به همین ترتیب انواع داده‌ی مختلف و تعریف و مقداردهی متغییرها تعریف می‌شود. در صورتی که خطای گرامری در برنامه وجود داشته باشد برنامه خارج می‌شود و هیچ‌گونه برنامه‌ی خروجی تولید نمی‌شود. هم‌چنین خط و شماره‌ی کاراکتر محل خطا چاپ می‌شود.

۳ analyzer Semantic

در بخش بررسی معنایی کد لازم است دقیقا به محتوای هر یک از توکن‌ها (Lexeme) توجه شود . در این بخش لازم است تعریف شدن متغییرها و محدوده‌ی مجاز استفاده از متغییرها با توجه به Scope ها تعریف کنید. مثلا باید بررسی شود که اگر یک متغییر در یک بلوک کد دو بار تعریف شده است خطای معنایی نمایش داده شود.

۴ ایده‌های کلی استفاده شده در پیاده‌سازی

برای ذخیره‌ی محدوده‌ی تعریف هر متغییر از تعدادی Hash-table که با اشاره‌گرهایی به جداول قبلی تعریف شده است استفاده می‌کنیم.
تمام متغییرهای محلی تعریف شده در توابع درون پشته ذخیره می‌شوند و با استفاده از ثبات‌های sp , fp استفاده می‌شود و برای هر فراخوانی تابع لازم است ثبات‌های مهم از اجرای تابع فراخوانی کننده ذخیره شود تا در زمان بازگشت قابل بازیابی و مقداردهی مجدد باشد.
برای رجوع و دسترسی به نزدیک‌ترین حلقه‌ها در اجرای کد از پشته‌ای برای حلقه‌های و Switch-case ها استفاده می‌شود و همیشه نزدیک‌ترین حلقه در بالای پشته قرار دارد.

۵ امکانات پیاده سازی شده

در این پروژه موارد زیر پیاده‌سازی شده است:

- تعریف متغییرهای جهانی
- تعریف توابع با حداکثر ۳ پارامتر ورودی
- توابع بازگشتی
- ساختارهای کنترلی for , while , switch-case , if-else
- اجرای دستورات break , continue
- تعریف مجدد متغییر هم‌نام در بلوک جدید
- ایجاد تعداد دلخواه بلوک کد تو در تو
- آرایه‌ی یک‌بعدی

کامپایلر در صورت وجود خطاهای زیر در برنامه‌ی ورودی با چاپ پیغام خطای مناسب خارج می‌شود و هیچ کدی تولید نمی‌شود.

- تعریف متغییر از نوع void
- تعریف مجدد یک متغییر در داخل یک بلوک کد
- دسترسی و scope نادرست یا قبل از تعریف متغییر
- دستور break خارج از حلقه‌ها و case-switch
- دستور continue خارج از حلقه
- فراخوانی توابع تعریف نشده
- فراخوانی تابع با تعداد آرگومان‌های نامناسب
- تعریف تابع با بیش از سه آرگومان
- در صورتی که تابع‌های غیر void مقدار خروجی برنگردانند.
- در صورتی که تابع‌های void مقدار خروجی داشته باشند.
- استفاده از متغییر به جای آرایه و با اندیس‌دهی

۶ تبدیل تحلیل‌گر لغوی و تحلیل نحوی به برنامه‌ی اجرایی

برای این کار یک فایل Makefile نوشته شده است که با دستورات و با بخش‌های مختلف فایل محل فایل ورودی و فایل نهایی که باید تولید شود را می‌گیرد و در صورتی که برنامه درست باشد کد خروجی به زبان mips تولید می‌شود در غیراینصورت خطا مورد نظر در خروجی کنسول چاپ می‌شود.