پروژه درس اصول طراحی کامپایلر(فاز دوم)

دکتر امین طوسی

دستیاران آموزشی: نجمه حبیبی، نسترن احمدی بنکدار

مهلت تحویل: 1400/2/15

مقدمه

در این فاز به پیاده سازی تحلیل گر معنایی و سپس به پیاده سازی دو خطا می پردازیم. قبل از اینکه خطا های برنامه را توسط تحلیل گر معنایی بررسی کنیم باید اطلاعاتی را جمع آوری و در جدول علائم ذخیره کنیم.

توضیحات

- جدول علائم (Symbol Table):

ساختار داده ای است که در کامپایلرها برای نگه داری و جمع آوری شناسه های تعریف شده در کد ورودی استفاده میشود.

- طراحی جدول علائم:

طراحی این جدول به شکل های متفاوتی قابل انجام است که با توجه به نوع زبان، پیچیدگی و نظر طراح کامپایلر تعیین میشود. ساده ترین نوع پیاده سازی یک جدول علائم استفاده از hashtable میباشد به این صورت که کلید آن نام شناسه و ارزش نسبت داده شده به این کلید مجموعه مقادیر ویژگی های مربوط به شناسه می باشد.

هر جدول علائم دو متد اصلی دارد که اطلاعات مربوط به شناسه از طریق این دو متد در جدول ذخیره و یا از آن بازیابی میشوند.

```
insert (idefName, attributes)
lookup (idefName)
```

هر جدول علائم دو متد اصلی دارد که اطلاعات مربوط به شناسه از طریق این دو متد در جدول ذخیره و یا از آن بازیابی میشوند. هر حوزه یک جدول علائم مخصوص به خود دارد.

حوزه ها (scopes):

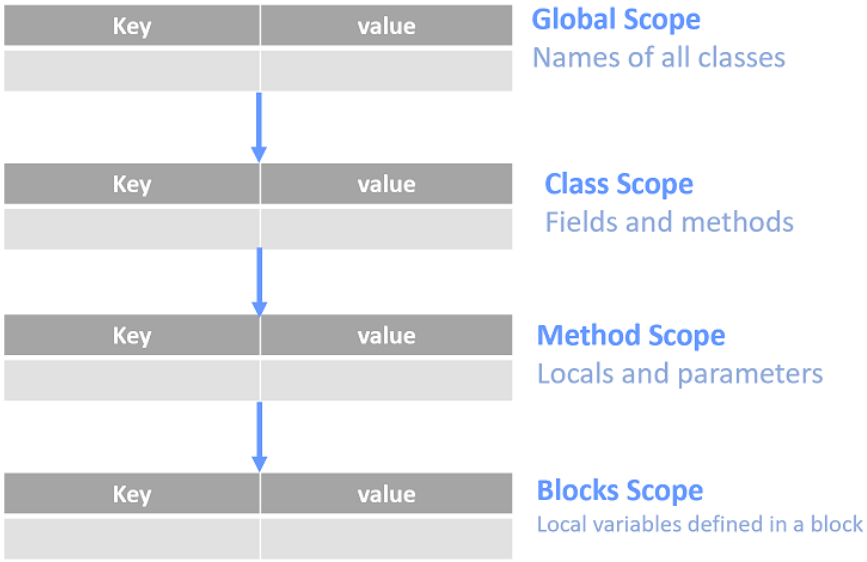
هر یک از موارد زیر زبان Cool یک حوزه به حساب می آیند.

- تعریف برنامه
- تعریف کلاس
- تعریف توابع
- شروع if , else, while و nestedStatement ها

- ارتباط حوزه ها و جداول علائم

هر حوزه یک جدول علائم دارد. لذا شناسه هایی که در هر حوزه تعریف می شوند باید در جدول علائم مربوط به این حوزه ذخیره شوند. از آنجایی که این زبان از حوزه های تودرتو پشتیبانی می کند یک روش مناسب برای نگهداری این حوزه ها استفاده از گراف می باشد .می توانید از ساختار ها ی دیگر هم استفاده نمایید .

ساختار جدول علائم برای حوزه ها به این صورت است:



حال که با جدول علائم آشنا شدیم، ابتدا با توجه به توضیحات داده شده، جدول علائم را ساخته و در گام بعد با استفاده از آن به بررسی خطا های موجود در برنامه که در ادامه توضیح داده شده است می پردازیم.

فرمت گزارش خطا

خطا های موجود در برنامه را بر اساس فرمت زیر گزارش دهید:

line شماره خط ارور و column پوزیشن آن را در یک خط نشان میدهد.

در این فاز شما به پیاده سازی دو نوع خطایی که در ادامه نشان داده شده است می پردازید.

1. **خطای تعریف دوباره کلاس/متد/خصیصه**

تعریف دوباره یک کلاس:

```
Error101 : in line  [line:column], class [name] has been defined already
```

تعریف دوباره متد در یک کلاس:

```
Error102 : in line  [line:column] , method  [name] has been defined already
```

تعریف دوباره یک خصیه در یک کلاس:

```
Error104 : in line [line:column], field [name] has been defined already
```

- نکته: دو نوع متفاوت میتوانند هم نام باشند به عنوان مثال اگر یک فیلد و متد هم اسم باشند مشکلی نیست.
- نکته: در صورت تعریف دوباره یک کلاس، متد ویا فیلد اسم آن را عوض میکنیم و به سیمبل تیبل اضافه میکنیم و اسم آن را به این صورت ذخیره میکنیم: name_line_column . بعنوان مثال اگر متغیر d دوباره تعریف شود آن را به صورت d_34_48 ذخیره می نماییم.
- نکته: هر کدام از موارد ذکر شده اگر دوبار تعریف شوند مورد دوم مطرح نیست و فرض میکنیم اصلا وجود ندارد و تنها از مورد اول استفاده میشود. به عنوان مثال اگر یک کلاس دوبار تعریف شده باشد تنها میتوان از کلاس اول استفاده کرد.

2. **خطای استفاده از متغیر/کلاس تعریف نشده**

استفاده از یک کلاس تعریف نشده در یک حوزه:

```
Error105 : in line [line:column], cannot find  class [className]
```

استفاده از یک متد تعریف نشده:

```
Error105 : in line [line:column], cannot find  method [methodName]
```

استفاده از یک متغیر یا خصیصه تعریف نشده:

```
Error106 : in line [line:column], Can not find Variable [name]
```

شاد و پیروز باشید.