الجلسة الثامنة

التعامل مع التصريحات Declaration إضافة لحالات خاصة من Semantic Actions

الهدف من الجلسة

- التعرف على كيفية تعديل بنية المترجم لإضافة القواعد الخاصة بتعليمات التصريح في البرنامج المصدري.
- التعرف على كيفية بناء Semantic Actions لتوليد استجابات محددة لبعض نتاجات الإعراب

مستلزمات الجلسة

- حاسب بمواصفات دنيا RAM: 1 GB, CPU: 1.6 GHz, Windows 10 OS 64 bit
- Turbo c++/ DevC++
- LEX & BISON tools

خطوات العمل

- بناء ملفي وصف ماسح ومعرب وتعلم إضافة بعض Semantic Actions عند الحاجة.

الخلاصة والنتائج:

يفترض عند نهاية الجلسة:

- تمكن الطالب من معرفة كتابة ملف وصف الماسح وملف وصف المعرب لمترجم يتضمن جزء تصريح قبل التعليمات الأساسية إضافة للتعامل مع الأخطاء وعبارات Semantic Actions معبرة عن حالات محددة مطلوبة.

1.1 إضافة جزء الإسناد للبرنامج المصدري

لتسهيل التعامل سنقوم بالعودة لمثال الجلسة السابقة ونقوم بالتعديل عليه:

المطلوب تعديل بنية ملفي وصف الماسح والمعرب بحيث يستطيعان التعرف على جزء التصريح في البرنامج مثلا:

بفرض أضفنا الجزء التالي لملف input.txt

```
int c=0;
switch(s)
{
case 1:a=b+c;break;
case 2:    if(a<0.5)
    then a=b*c;
    break;
case 3:    while(a5)
    b=b+a;
    break;
}
```

الجزء الذي أضفناه يتضمن تعليمة تصريح عن متحول اسمه c نوعه integer وقيمته الابتدائية 0.

1.2 التعديلات في بنية ملف وصف الماسح

نضيف فقط القوالب اللازمة للتعرف على أنماط البيانات مثلاً سنضيف التعرف على نمط البيانات int ونمط double

```
int                return DEC_INT;

double                    return DEC_DOUBLE;
```

1.3 التعديلات في بنية ملف وصف المعرب

نعدل أولاً بإضافة Tokens الخاصة بالأنواع INT و DOUBLE لقسم التصريح عن Tokens

%token ID NUM REAL IF THEN LE GE EQ NE OR AND ELSE EQUAL LT GT PLUS MINUS MULT DIVS LPAR RPAR LBRAC RBRAC POINTS SEMI SWITCH CASE DEFAULT WHILE BREAK **DEC_INT DEC_DOUBLE**

والآن نعدل قواعد الإعراب بحيث تقبل التعرف على وجود قسم تصريح في بداية البرنامج كالآتي:

S : DecPart SWITCH_Part S {if (errors==0) printf("Input accepted.\n");}| ;

DecPart : Type ID EQUAL NUM SEMI;

Type: DEC_INT| DEC_DOUBLE;

SWITCH_Part : ……..

نتابع باقي القواعد كما هي.

نلاحظ أننا أضفنا في البداية قاعدة تفيد أن البرنامج المصدري هو قسمان قسم تصريح DecPart وقسم آخر خاص بحلقة Switch.

بعد ذلك نعرف بنية جزء DecPart وهو عبارة عن 1- نوع 2- اسم متغير 3- إشارة مساواة 4- رقم 5- فاصلة منقوطة.

بعد ذلك نعرف Type والذي يمكن أن يكون int أو double.

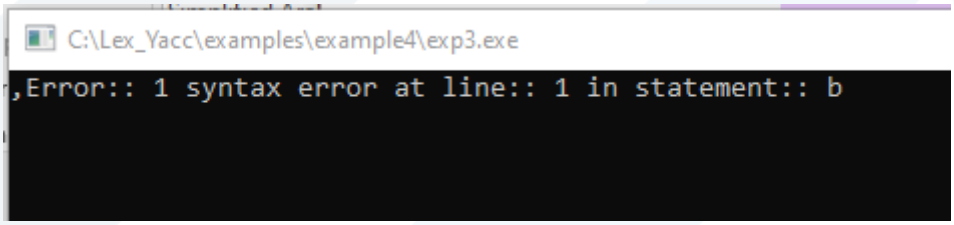
والآن نختبر المترجم:

لنغير محتوى ملف input.txt ليصبح:

```
int c,b=0;
switch(s)
{
```

```
case 1:a=b+c;break;
case 2:    if(a<0.5)
    then a=b*c;
    break;
case 3:    while(a5)
    b=b+a;
    break;
}
```

الخطأ هو أننا عرفنا متغير آخر مع نفس المتغير c ونلاحظ أن المترجم أعطى خطأ رغم أن هذه العملية مسموحة في كثير من لغات البرمجة.



ما الحل برأيك ليصبح المترجم قادراً على التعرف على هذه الحالة من البرنامج المصدري.

1.4 التعامل مع Semantic Actions

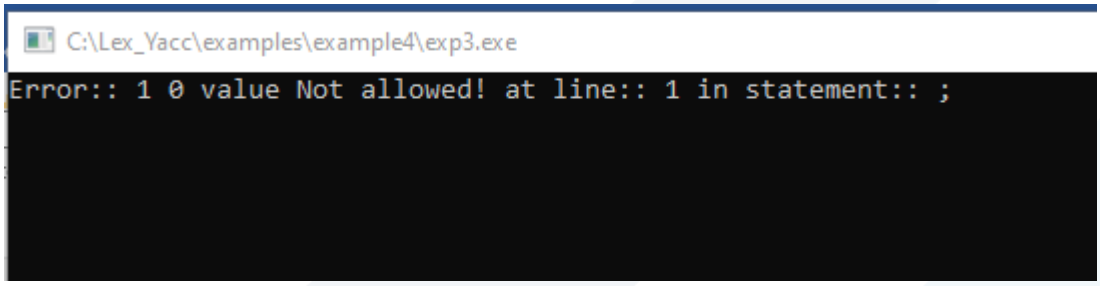
بفرض أننا نريد من المترجم أن يمنع المستخدم من إعطاء قيمة الصفر لمتغير ما خلال مرحلة التصريح.

أي مثلاً عندما يكون محتوى ملف input.txt هو:

```
int c=0;
switch(s)
{
case 1:a=b+c;break;
case 2:    if(a<0.5)
    then a=b/c;
    break;
case 3:    while(a5)
    b=b+a;
    break;
```

}

يعطي المترجم خطأ.



طريقة الحل: نعدل ملف وصف المعرب وتحديداً القاعدة الخاصة بقسم التصريح فقط.

DecPart : Type ID EQUAL NUM SEMI {if ($4==0) yyerror("0 value Not allowed!");};

حيث هنا تم استدعاء تابع الخطأ برسالة خطأ عندما تكون قيمة المؤشر $4 وهي قيمة المتحول المصرح عنه مساوية للصفر.

ارسم شجرة الإعراب للعبارة المصدرية:

```
int c=0;
switch(s)
{
case 1:a=b+c;break;
case 2:    if(a<0.5)
    then a=b/c;
    break;
}
```

انتهت الجلسة – د. علي ميا ، م. رشا شباني