الجلسة الأولى: تصميم مترجمات

الهدف من الجلسة

- التعريف بمفهوم المترجمات والمفسرات.
- التعريف بمراحل التحليل اللفظي والقواعدي والمعنوي وتوليد الشيفرة.

المترجمات compiler

برنامج يقوم بتحويل لغة مصدرية إلى لغة هدف.
أشهر أمثلتها مترجم يقوم بتحويل لغة عالية المستوى مثل C أوjava إلى لغة assembly خاصة بالحاسب.

يوجد نوع أخرمن البرامج المستخدمة في ترجمة لغات البرمجة وهي المفسرات interpreters تختلف عن المترجمات في ألية العمل فالمترجمات تقول بترجمة الكود المصدري كاملة ثم تنفيذه في حين أن المفسرات تقوم بترجمة الكود سطراً سطراً ثم تنفيذ كل سطرمباشرة بعد ترجمته.
أشهر أنواع المفسرات، مفسر لغة الـPython.

مراحل الترجمة

تمر عملية الترجمة بالمراحل التالية:

مرحلة تحليل المفردات:

يقوم بها المحلل اللفظيlexical analyzer حيث يقوم بتحويل سلسلة المفردات إلى سلسلة من الرموز tokens.

مثال:

السطر البرمجي التالي:

'print 3 + 1'

سينتج عنه مجموعة الرموز:

['print','1','+','3']

في الواقع، كل رمز سيحتوي نوع وقيمة موافقة (يتحدد في جدول الرمز).

يتم تحديد نوع الرمز وقيمته بناءً على قوالب محددة مسبقاً تم تعريف المترجم عليها تدعى بالتعابير النظامية regular expressions.

يتم تخزين المعلومات الأساسية في جدول الرموز:

مثال:

ليكن لدينا الكود البرمجي التالي الموافق لغة pascal:

```
For     i:= 1 to vmax  do  a :=  a+i ;
```

نوع المفردة Tokens	ماهيتها
كلمة محجوزة	For
معرف	I
إسناد	:=
عدد صحيح	1
كلمة محجوزة	To
معرف	Vmax do
كلمة محجوزة	يخ
معرف	A
إسناد	:=
عملية رياضية	+
معرف	I
فاصلة نهاية التعليمة	;

ثم يلي ذلك مرحلة بناء جدول الرموز(symbol table)، وهي عبارة عن قائمة من تركيبات تحمل خصائص المفردات Tokens التي تم تحديدها في الخطوة السابقة:

رقم الرمز	Token	نوع Token	نوع variable
1	For	كلمة محجوزة	-
2	For	كلمة محجوزة	-
3	For	كلمة محجوزة	-
4	;	فاصل	-

5	I	معرف	Integer

إذا تمثل هذه المرحلة مسحاً Scanning لتسلسل الدخل لتحديد المفردات Tokens الداخلة في تركيب العبارة، ولا يهتم المحلل اللفظي بترتيب الـ Tokens وبالتالي السطر التالي صحيح تماما بالنسبة له:

For for for i:= := 10 do for a a;

تعد هذه العبارة صحيحة لفظياً lexically correct لكنها خاطئة قواعدياً، وهنا تأتي مرحلة التحليل القواعدي لتحديد هذه الأخطاء.

مرحلة التحليل القواعدي Syntax Analysis

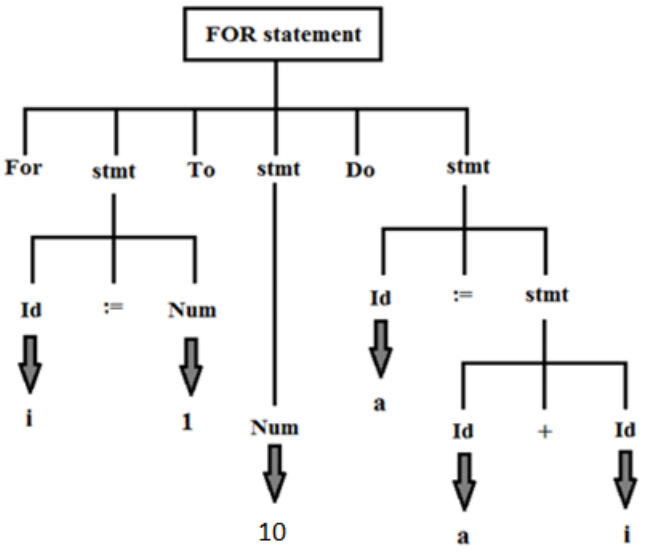
يتم خلال هذه المرحلة التحقق من النحو الخاص باللغة ويكون هذا وفقا لمجموعة من القواعد والتي تمثل الضابط الذي يحكم صحة تسلسل المفردات Tokens التي تم مسحها من قبل المحلل اللفظي.

يقوم المحلل القواعدي ببناء الشجرة باستخدام مجموعة Tokens التي تم توفيرها بالجدول السابق، مع الأخذ بعين الاعتبار قواعد اللغة التي ستحدد فيما إذا كانت العبارة الممسوحة صحيحة قواعدياً أو لا.

قد تكون العبارة صحيحة لفظياً وقواعدياً، لكنها ليست ذات معنى.

شجرةالإعراب للكود التالي:

for i:=1 to 10 do a:=a+i



مرحلة تحليل المعاني Semantic Analysis:

تعتمد هذه المرحلة بشكل أساس على جدول الرموز المنشأ سابقاً والذي يتضمن أسماء الرموز التي صادفها الماسح خلال مسحه لسلسلة الدخل.

هناك العديد من الشروط التي يتم التأكد منها في هذه المرحلة مثل:

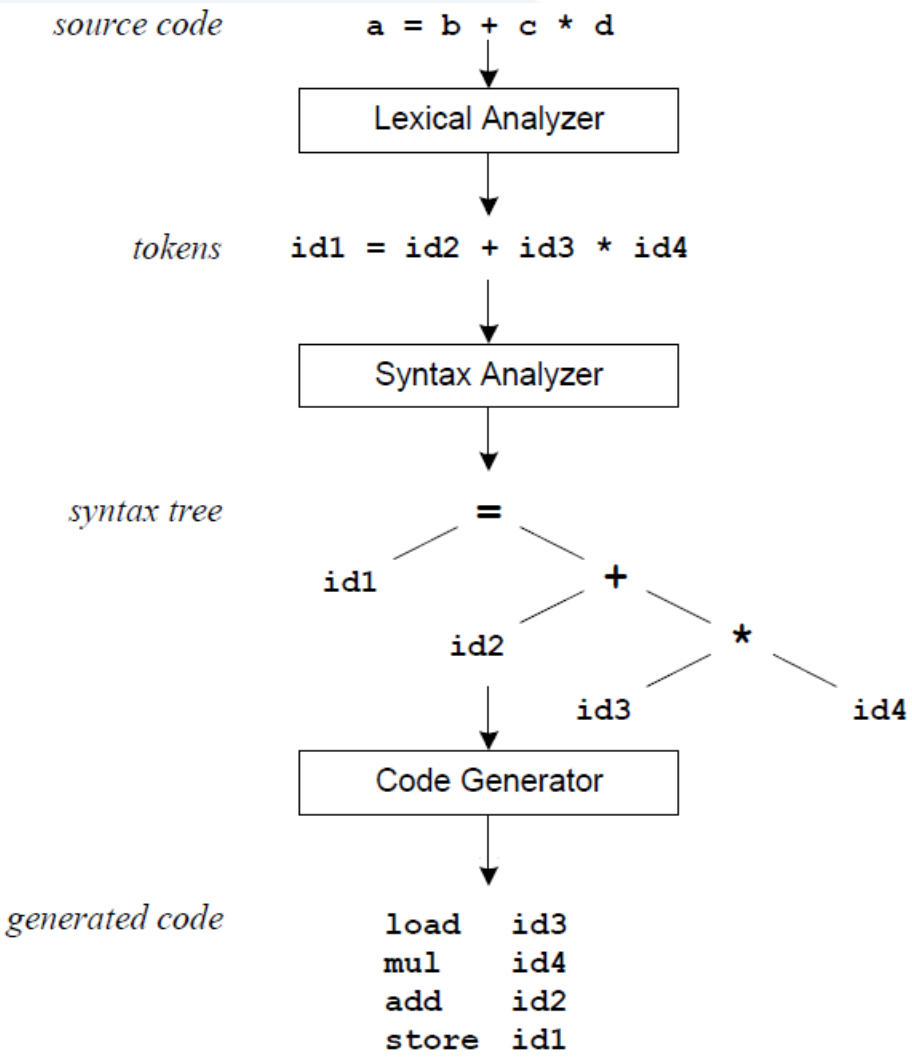
- التأكد من تطابق أنواع المتغيرات مثل إسناد رقم حقيقي real لمتغير من النوع الصحيح integer.
- التأكد من تطابق نوع المتغير مع قيمته.
- التأكد من عدم تكرار اسم متغير معرفاً مسبقاً
- التأكد من عدم إسناد قيمة لمتغير غير مصرح عنه سابقاً

مرحلة توليد الكود Code generation

وهنا نقوم بعملية إنتاج كود بلغة الآلة أو لغة التجميع.

مثال بسيط: ماهي مراحل الترجمة اللازمة لترجمة العبارة التالية:

A=B+C*D



الأدوات الأساسية لبناء المترجمات

يوجد عدد من الأدوات البرمجية التي يمكن استعمالها لبناء المترجمات ولكن يوجد هناك أداتين أساسيتين هما الأكثر شهرة لذلك هما:

- **LEX**
- **BISON**

الأداة البرمجية (LEX):

وهي أداة تقوم بتوليد محلل مفردات أو ما يتم تسميته عادة بـ Scanner مكتوب بلغة C حيث يتم استخدام قوالب جاهزة لمطابقة السلاسل المحرفية الموجودة في الكود المصدر وتحويلها إلى tokens

مثلاً عندما يصادف الماسح (Scanner) المتغير x في سلسلة الدخل فإنه يرسل المفردة (token) التي تمثل هذا المتغير إلى المحلل القواعدي مثلاً <id> ، كما يرسل له أيضا اسم المتغير ويقوم بإدخاله إلى جدول الرموز وتعيين خصائصه كنوعه

وقيمته وغيرها من الخصائص.

الأداة البرمجية (BISON or YACC)

تولد هذه الأداة شيفرة بلغة C لمحلل قواعدي ويعرف أيضا بالمعرب (parser). يستعمل المعرب قواعد اللغة لتحليل الـ Token القادمة من الـ Scanner ويبني منها شجرة الإعراب الموافقة لتسلسل الـ Token.

في حالة عدم مطابقة العبارة للنحو، يصدر الـ BISON رسالة خطأ اعتماداً على تابع الخطأ المزود به (سيتم استعراضه لاحقاً عند شرح بنية ملف parser).

سنقوم في هذا المقرر باستخدام لغة البايثون مع مكتبة PLY وهو تطبيق في لغة البايثون لأداة LEX وYACC وهو يتكون من نموذجين منفصلين lex.py لعملية المسح وتحليل المفردات.

ونموذج yacc.py لعملية الإعراب.

انتهت الجلسة