# أمثلة المحلل اللغوي (Parser Examples)

## مراجعة البنية النحوية

---

## 📋 قواعد النحو (Grammar)

### 💬 ملاحظة عن التعليقات

لن يتلقى أي Parser هذا يعني أن الـ .(Lexer) التعليقات )التي تبدأ بـ `//` وتستمر حتى نهاية السطر( يتم التعامل معها وتجاهلها بالكامل بواسطة المحلل اللغوي رموز (Tokens) تمثل التعليقات، وبالتالي يمكن كتابة التعليقات في أي مكان في الكود دون التأثير على بنية الجملة النحوية.

---

## 🔹 القواعد النحوية الأساسية

### البرنامج (Program)

```
Program → ClassList EOF
```

### تعريف العيلة (Class Definition)

```
ClassDefinition → عيلة Identifier { FunctionList }
```

### قائمة الكلاسات (ClassList)

```
ClassList → ClassDefinition ClassList | ε
```

### قائمة الدوال (FunctionList)

```
FunctionList → FunctionStmt FunctionList | ε
```

> ***ملاحظة مهمة:*** *يجب أن تحتوي قائمة الدوال في إحدى الكلاسات على دالة واحدة فقط اسمها* ***ساموعليكم*** *(الدالة الرئيسية).* ⚠️

---

## 🔸 تعريف الدوال والمعاملات

### تعريف الدالة (FunctionStmt)

```
FunctionStmt → [جاعد] حيسبة [Type] Identifier (ParameterList) BlockStmt
```

**حيث:**

- `Identifier` هو اسم الدالة

**حيث:**

- `Identifier` هو اسم الدالة
- **الدالة الرئيسية:** يجب أن تكون دالة واحدة فقط باسم **ساموعليكم**
- تشير إلى `جاهز` `static`

### قائمة المعاملات (ParameterList)

```
ParameterList → [Parameter (, Parameter)*]
Parameter → Type Identifier
```

> 📝 ***ملاحظة:*** *في الكود البرمجي، تم افتراض أن المعاملات يجب أن تكون من نوع* ***رقم*** *(TokenType.Rakam) لتبسيط عملية التحليل. يمكن توسيعها لتشمل أنواعًا أخرى لاحقًا.*

---

## 🔹 الكتل والجمل البرمجية

### الكتلة البرمجية (BlockStmt)

```
BlockStmt → { StatementList }
```

### قائمة الجمل (StatementList)

```
StatementList → Statement StatementList | ε
```

### الجملة (Statement)

```
Statement → Declaration
          | Assignment
          | IfStmt
          | WhileStmt
          | ForStmt
          | PrintStmt
          | ReturnStmt
          | BlockStmt
```

---

## 🔸 أنواع الجمل البرمجية

### التصريح عن متغير (Declaration)

```
Declaration → Type Identifier [= Expression] ;
```

**حيث:**

```
Type → صح_غلط | كسر | كلام | رقم
```

### الإسناد (Assignment)

```
Assignment → Identifier = Expression ;
```

### الإسناد (Assignment)

```
Assignment → Identifier = Expression ;
```

### جملة الشرط (IfStmt)

```
IfStmt → لو (Condition) Statement [وإلا Statement]
```

### جملة التكرار (WhileStmt)

```
WhileStmt → علطول (Condition) Statement
```

### جملة التكرار (ForStmt)

```
ForStmt → لف ([Declaration | Expression] ; [Condition] ; [Expression]) Statement
```

### جملة الطباعة (PrintStmt)

```
PrintStmt → اكتب Expression ;
```

### جملة الإرجاع (ReturnStmt)

```
ReturnStmt → الجوف [Expression] ;
```

---

## 🔹 التعبيرات (Expressions)

### التعبير (Expression)

```
Expression → Comparison
```

### المقارنة (Comparison)

```
Comparison → Term (RelOp Term)*
```

**حيث:**

```
RelOp → == | != | > | >= | < | <=
```

### الحد (Term)

```
Term → Factor (AddOp Factor)*
```

**حيث:**

```
AddOp → + | -
```

```
AddOp → + | -
```

### العامل (Factor)

```
Factor → Postfix (MulOp Postfix)*
```

**حيث:**

```
MulOp → * | /
```

### الزيادة اللاحقة (Postfix)

```
Postfix → Primary [++]
```

### الأساسي (Primary)

```
Primary → Literal | Identifier | (Expression)
```

**حيث:**

```
Literal → NumberLiteral | StringLiteral
```

## 📌 ملاحظات إضافية

### 🎯 الدالة الرئيسية (Main Function)

يجب أن يحتوي البرنامج على دالة واحدة فقط باسم **ساموعليكم** (SamoAlikom) لتمثل نقطة الدخول (Entry Point) للبرنامج، ويجب أن تكون هذه الدالة ضمن إحدى الكلاسات المعرفة.

---

## 🌳 هيكل شجرة بناء الجملة المجردة (AST Nodes)

العقد التالية (AST) لتمثيل البنية الجديدة متعددة الكلاسات، يجب أن تتضمن شجرة بناء الجملة المجردة:

| العقدة | الوصف |
|---|---|
| **ProgramNode** | يمثل البرنامج بالكامل، ويحتوي على قائمة الكلاسات |
| **ClassNode** | يمثل تعريف العيلة الواحدة |
| **FunctionNode** | يمثل تعريف الدالة |
| **VarDeclNode** | التصريح عن متغير (Declaration) |
| **AssignmentNode** | جملة الإسناد |
| **PrintNode** | جملة الطباعة (PrintStmt) |
| **IfNode** | جملة الشرط (IfStmt) |
| **WhileNode** | جملة التكرار (WhileStmt) |

| | |
|---|---|
| **IfNode** | جملة الشرط (IfStmt) |
| **WhileNode** | جملة التكرار (WhileStmt) |
| **ForNode** | جملة التكرار (ForStmt) |
| **ReturnNode** | جملة الإرجاع (ReturnStmt) |
| **BlockNode** | الكتلة البرمجية (BlockStmt) |
| **BinaryOpNode** | التعبيرات الثنائية )مثل الجمع والمقارنة( |
| **LiteralNode** | القيم الثابتة |
| **VariableNode** | المتغيرات |
| **PostfixIncrementNode** | عملية الزيادة اللاحقة |

---

## 📚 خلاصة

هذا المستند يحدد القواعد النحوية الكاملة للغة البرمجة، بدءًا من البرنامج الكامل وصولاً إلى أصغر التعبيرات. يمكن استخدام هذه القواعد لبناء محلل لغوي (Parser) قادر على تحليل الكود المكتوب بهذه اللغة وبناء شجرة AST المناسبة.