Cho luật sinh của lệnh gán trong văn phạm của một ngôn ngữ như sau:

assign → ID EQUAL exp SEMI

trong đó ID là token đại diện cho một danh hiệu, EQUAL là token đại diện cho dấu gán '=', exp là ký hiệu không kết thúc đại diện cho biểu thức và SEMI là token đại diện cho dấu ';'. Ngữ nghĩa của lệnh gán là lấy giá trị của biểu thức exp gán vào biến có tên là ID. Giả sử trên cây AST, nút ASSIGN dùng để biểu diễn lệnh gán. Vậy nút này sẽ cần phải có bao nhiêu nút con?

Hint

- ○ 4
- ○ 3
- ◉ 2
- ○ 1

Chính xác, nút ASSIGN chỉ cần thêm 2 thông tin là đủ: biến nào được gán và được gán với giá trị nào. Do đó, nút ASSIGN chỉ cần 2 nút con là đủ

Chọn cây AST thích hợp nhất cho biểu thức: 12 * (3 + 5) - 20 (độ ưu tiên các phép toán như thông lệ)?

- ○ BinExp("*",Lit(12),BinExp("-",BinExp("+",Lit(3),Lit(5)),Lit(20)))
- ○ BinExp("-",Lit(12),BinExp("*",BinExp("+",Lit(3),Lit(5)),Lit(20)))
- ○ BinExp("-",Lit(20),BinExp("*",Lit(12),BinExp("+",Lit(3),Lit(5))))
- ◉ BinExp("-",BinExp("*",Lit(12),BinExp("+",Lit(3),Lit(5))),Lit(20))

Một giải pháp khác để biểu diễn biểu thức nhị phân là thay vì định nghĩa lớp BinOp như trong bài giảng, ta định nghĩa các lớp Plus, Minus, Mul and Div tương ứng cho các phép toán +,-,* và /. Trong trường hợp này, một khai báo lớp Plus sẽ như thế nào?

- ○ case class Plus(op:String,e1:Exp,e2:Exp) extends Exp
- ○ case class Plus(e1:Exp,e2:Exp)
- ◉ case class Plus(e1:Exp,e2:Exp) extends Exp
- ○ case class Plus(e:Exp) extends Exp

Giả sử ta có luật sinh sau trong văn phạm:

assign → ID ASSIGN exp SEMI

Trên cây phân tích cú pháp (parse tree) sinh ra bởi ANTLR, nút ứng với luật này là đối tượng của lớp có tên là?

- ○ assign
- ○ Assign
- ◉ AssignContext

Cho luật sinh của một văn phạm được viết ở dạng EBNF như sau:

prog → stmt+

Số nút con của nút ứng với prog trên cây phân tích cú pháp là bao nhiêu?

- ○ 0
- ○ 1
- ◉ Bất cứ số dương (>0) nào.

Cho luật sinh của văn phạm được viết như sau:

expp → ADD term expp | ∈

Số nút con của nút ứng với expp là bao nhiêu?

- ☑ 0

  Correct
- ☐ 1

  Incorrect
- ☐ 2

  Incorrect
- ☑ 3

  Correct

Hide Feedback

Hai đáp án đúng là 0 và 3. Số 0 ứng với trường hợp vế phải là ∈, trong khi số 3 ứng với trường hợp vế phải là **ADD term expp**.

Cho luật sinh của một văn phạm như sau:

assign → ID ASSIGN exp SEMI

Giả sử biến **ctx** đang cất giữ nút ứng với assign trên cây phân tích cú pháp sinh ra bởi ANTLR. Hãy viết biểu thức để truy xuất nút ứng với **exp** bên vế phải?

- ○ ctx.ExpContext
- ○ exp()
- ○ ctx.getChild(2)
- ◉ ctx.exp

Cho luật sinh của một văn phạm như sau:

ifstmt → IF exp THEN stmt ELSE stmt

Giả sử biến **ctx** đang cất giữ nút ứng với **ifstmt** trên cây phân tích cú pháp (parse tree). Hãy viết biểu thức để truy xuất nút ứng với **stmt** của vế else?

- ctx.stmt
- ctx.stmt(2)
- ctx.stmt(1)
- ctx.getChild(5)

Cho luật sinh của văn phạm như sau:

prog → stmt+

Giả sử ctx là biến đang giữ nút ứng với prog. Hãy viết biểu thức để nhận được một danh sách tất cả các nút con **stmt** của nút ứng với **prog**?

- stmt
- ctx.stmt
- ctx.stmt(0)
- ctx.getChildren()

Cho văn phạm có các luật sinh sau:

exp → exp ADD term | term

term → term MUL fact | fact

fact → ID | INTLIT | LP exp RP

Với văn phạm trên, ANTLR sẽ sinh ra lớp Visitor có bao nhiêu phương thức visit chủ yếu ?

- 7
- 3
- 6
- 9

Cho văn phạm của ngôn ngữ ABC có luật sinh sau:

exp → exp ADD term | term

Prototype của phương thức visit sinh ra bởi ANTLR cho luật sinh này là gì?

- T visitExpContext(ABCParser.Exp ctx)
- T visitExp(Exp ctx)
- T visitExp(ABCParser.ExpContext ctx)
- T visitExpContext(ABCParser.ExpContext ctx)

Cho các luật sinh của văn phạm của ngôn ngữ ABC như sau:

exp → exp ADD term | term

term → term MUL fact | fact

fact → ID | INTLIT | LP exp RP

trong đó ADD là token đại diên phép toán +; MUL đại diện *; ID cho danh hiệu; INTLIT cho số nguyên, LP cho { và RP cho }

và cấu trúc dữ liệu cho cây AST được định nghĩa (trên ngôn ngữ Scala) qua các lớp sau:

```
trait ExpAST
case class Plus(e1:ExpAST,e2:ExpAST) extends ExpAST
case class Mul(e1:ExpAST,e2:ExpAST) extends ExpAST
case class Ident(id:String) extends ExpAST
case class Intlit(val:Int) extends ExpAST
```

Cho biết đối tượng Plus (lớp con của ExpAST) sẽ được tạo ra trong phương thức visit nào?

- ○ Plus được tạo ra trong phương thức visitTerm
- ◉ Plus được tạo ra trong phương thức visitExp
- ○ Plus được tạo ra trong phương thức visitFact

Chính xác, token ADD đại diện cho phép + nên Plus được tạo ra trong visitExp

Cho biết đối tượng Mul (lớp con của ExpAST) sẽ được tạo ra trong phương thức visit nào?

- ◉ Mul được tạo ra trong phương thức visitTerm
- ○ Mul được tạo ra trong phương thức visitExp
- ○ Mul được tạo ra trong phương thức visitFact

Chính xác, phương thức visitTerm có thể tạo ra Mul khi các nút con của nó ứng với vế phải term MUL fact.

Cho biết đối tượng Id (lớp con của ExpAST) sẽ được tạo ra trong phương thức visit nào?

- ○ Id được tạo ra trong phương thức visitExp
- ○ Id được tạo ra trong phương thức visitTerm
- ◉ Id được tạo ra trong phương thức visitFact
- ○ Id được tạo ra trong phương thức visitID

Đúng, Id được tạo ra trong phương thức visitFact khi nút con của nó là ID